NATURAL Version 2.3.1
Release Notes for Mainframes

Manual Order Number: NAT231-008IBB

This document applies to NATURAL Version 2.3.1 for mainframes, and to all subsequent releases.

Specifications contained herein are subject to change, and these changes will be reported in subsequent release notes or new manual editions.

TABLE OF CONTENTS

GENERAL INFORMATION

Introduction

These *Release Notes* inform you of the enhancements and new features that are provided with Version 2.3.1 of NATURAL.

Some of these enhancements lead to intentional minor incompatibilities between Version 2.2 and Version 2.3 (see below).

In addition to providing the enhancements and new features described in these *Release Notes*, NATURAL Version 2.3.1 also consolidates all error corrections, modifications and enhancements provided with the the previous system maintenance (SM) releases of Version 2.2.

All enhancements and new features described in these *Release Notes* are fully documented in the NATURAL Version 2.3.1 documentation set.

Prerequisites

NATURAL Version 2.3 requires the following versions of the following operating/ teleprocessing systems:

- BS2000 Version 10 (or above),
- MVS/XA Version 1 (or above),
- VSE/ESA Version 1.3 (or above),
- COM-PLETE Version 4.6 (or above),
- CICS/MVS Version 2.1 (or above),
- CICS/VSE Version 2.1 (or above),
- IMS/TM Version 3.1 (or above).

Compatibility

Applications that were created with NATURAL Version 2.2 can be executed with Version 2.3 without any conversion procedure being required, and without your having to make any adjustments to the programs — except in the few cases of intentional minor incompatibilities listed below.

Overview of Intended Incompatibilities

The following list provides an overview of the intentional incompatibilities introduced with Version 2.3 (for details on each of the topics listed, refer to the pages indicated).

When a Version 2.2 application is executed with Version 2.3, these incompatibilities will cause the application to produce "better", but slightly different, results. If in these cases you wish to get the same results as with Version 2.2, you have to adjust your applications accordingly.

- Redefinition of database arrays — a variable index range can no longer be specified in the redefinition of a periodic-group field or multiple-value field (see page 20).

- DEFINE WINDOW — the specification of too small a window size leads to a compilation error (see page 21).

- DIVIDE — with both GIVING and REMAINDER, different results for the REMAINDER field may occur (see page 22).

- FIND —the comparison logic for multiple-value fields in the WITH clause has been changed (see page 24).

- Incomplete statement blocks — no longer allowed (see page 30).

- Assignment of numeric value to alphanumeric field — only H'F0' truncated (see page 30).

- No upper-case translation for system variable *COM — AD=T for a *COM field will lead to a compilation error (see page 31).

- Computation of floating-point exponentiation corrected (see page 95).

- Results for SIN, COS and TAN functions (see page 95).

- More precise results for SQRT function (see page 96).

- Assignment of numbers with decimal digits to time corrected (see page 96).

- Assignment of negative numbers to date and time intercepted (see page 97).

- More precise results for arithmetic expressions with floating point operands (see page 97).

- More precise results for floating-point conversions (see page 98).

- Sign of packed numbers in assignments — different handling (see page 99).

- Assignments between numeric variables of the same length — different internal handling (see page 99).

- Interception of mismatching array ranges — incorrect results will lead to runtime error (see page 100).

- Comparison and assignment of variable array ranges — no longer allowed if an array range is actually a scalar (see page 100).

- Error messages NAT1117and NAT0924 replaced by NAT0082 (see page 102).

NATURAL System Files

Most NATURAL system files can be shared between Versions 2.2 and 2.3:

FNAT

A new FNAT system file is required for NATURAL Version 2.3.

It is strongly recommended that an empty ADABAS file be used as Version 2.3 FNAT file.

FUSER

The existing Version 2.2 FUSER system file can be shared by NATURAL Versions 2.2 and 2.3.

As a prerequisite, the following update INPLs must have been applied to your NATURAL Version 2.2 environment:

- NA2875,

- NT2801 (only if NATURAL Connection is installed),

- NE2841 (only if NATURAL SECURITY is installed),

- NQ3404 (only if NATURAL for DB2 is installed),

- NQ3405 (only if NATURAL for SQL/DS is installed).

When you apply these update INPLs, make sure that both FNAT and FUSER point to your NATURAL Version 2.2 environment. (The reason is that under Version 2.2, system programs were loaded with INPL into the library SYSTEM on the FUSER system file; these programs are not compatible with the Version 2.3 environment; with the above update INPLs, these programs are replaced by ones which can be executed under Versions 2.2 and 2.3.)

As of Version 2.3, objects to be loaded into library SYSTEM are no longer loaded into the system files FNAT and FUSER, but only into the FNAT system file (see also page 107). Also, library SYSTEM on the FNAT file is now the last steplib for user applications (see page 93).

FDIC

The existing Version 2.2 FSEC system file can be shared by NATURAL Versions 2.2 and 2.3.

If PREDICT is used in both NATURAL environments, the sharing of the system file requires that PREDICT Version 3.4.1 (or above) has been installed.

FSEC

The existing Version 2.2 FSEC system file can be shared by Versions 2.2 and 2.3 of NATURAL SECURITY.

As a prerequisite, update INPL NE2841 must have been applied to your NATURAL Version 2.2 environment.

FSPOOL

A new FSPOOL system file is required for NATURAL ADVANCED FACILITIES Version 2.3.

You can transfer the contents of your existing Version 2.2 FSPOOL system file to the new Version 2.3 FSPOOL system file by using the CONVERT utility provided by NATURAL ADVANCED FACILITIES Version 2.3.

Components/Interfaces Not Yet Available

The following components/interfaces of NATURAL are not yet available with Version 2.3.1:

- the debugging utility,

- the call statistics (application testpath statistics) utility,

- the NATURAL IMS/TM interface,

- NATURAL for VSAM under VSE,

- NATURAL for ADABAS SQL.

NATURAL for VSAM under VSE and NATURAL for ADABAS SQL will be made available as soon as possible after the release of Version 2.3.1.

The debugging utility, the call statistics utility and the NATURAL IMS/TM interface will be made available with Version 2.3.2 (which is scheduled for release in early 1998).

Other SOFTWARE AG Products with NATURAL 2.3

To use the following SOFTWARE AG products in conjunction with NATURAL Version 2.3, the following product versions (or above) are required:

Product	Version
ADABAS	5.3.3 (6.2.1 if NATURAL is to be used in a Sysplex environment)
ADABAS ONLINE SERVICES	6.2.1 or 6.1.*n* For Version 6.1.*n*, a ZAP has to be applied (which can be obtained from SOFTWARE AG Support).
ADABAS SQL Server AIF	1.4.2
ADABAS Stored Procedures And Triggers	6.2.2
ADABAS TEXT RETRIEVAL SYSTEM	2.1.3 A ZAP has to be applied (which can be obtained from SOFTWARE AG Support).
CON-NECT	3.3.1
ENTIRE BROKER ACI	2.1.2
ENTIRE EVENT MANAGEMENT	not yet decided
ENTIRE OPERATIONS	3.1.1
ENTIRE OUTPUT MANAGEMENT	1.4.1
ENTIRE REVIEW NATURAL MONITOR	3.5.1
ENTIRE SYSTEM SERVER	2.1.1
ENTIRE SYSTEM SERVER Interface	2.3.1
NATURAL ADVANCED FACILITIES	2.3.1
NATURAL CONSTRUCT	not yet decided
NATURAL DOCUMENT MANAGEMENT	not yet decided
NATURAL for ADABAS SQL	2.4.1
NATURAL for DB2	2.4.1
NATURAL for DL/I	2.3.1

Product	Version
NATURAL for VSAM	2.4.1
NATURAL for SQL/DS	2.4.1
NATURAL ISPF	2.3.1
NATURAL SECURITY	2.3.1
PREDICT	3.4.1
PREDICT APPLICATION CONTROL	2.2.1
SUPER NATURAL	not yet decided

Some of the above product versions are not yet available. For information on their availability, please contact your local SOFTWARE AG representative.

Documentation

A completely revised set of NATURAL manuals is provided with the release of NATURAL Version 2.3.1 for mainframe computers.

The basic NATURAL Version 2.3.1 documentation set for mainframe computers consists of the following manuals:

- *NATURAL Master Index for Mainframes*

- *NATURAL User's Guide for Mainframes* (*)

- *NATURAL Programming Guide* (* †)

- *NATURAL Reference Manual* (* †)

- *NATURAL Statements Manual* (* †)

- *NATURAL Utilities Manual for Mainframes* (*)

- *NATURAL Installation and Operations Manual for Mainframes*

(*) These manuals are also available in German.

(†) These manuals also apply to Version 3.1.1 of NATURAL on non-mainframe platforms, which will be released in due course.

For an overview of the contents of the manuals, please refer to the *NATURAL Master Index for Mainframes*. Also, the contents of each manual is summarized in its preface.

The manuals are available on CD-ROM as well as in printed form.

Example Library for New Features

The library SYSEXV23 contains several example programs which illustrate some of the new features of NATURAL Version 2.3.

PROGRAMMING LANGUAGE

This chapter contains information on:

- new statements,

- new system variables,

- enhanced system variable,

- new system function,

- enhanced statements,

- SQL.

New Statements

CLOSE CONVERSATION

The statement CLOSE CONVERSATION is used in conjunction with the NATURAL remote procedure call (see page 89). It enables the client to close conversations.

You can close:

- a specific conversation,

- the current conversation (as identified by the new system variable *CONVID), or

- all active conversations.

OPEN CONVERSATION

The statement OPEN CONVERSATION is used in conjunction with the NATURAL remote procedure call. It enables the client to open a conversation and specify the remote subprograms to be included in the conversation (see page 89).

New System Variables

The following new system variables are available:

System Variable	Function
*CONVID	Contains the conversation ID of a conversational RPC (see page 89).
*HARDWARE	Contains the name of the hardware platform on which NATURAL is running.
*MACHINE–CLASS	Contains the name of the machine class on which NATURAL is running.
*OS	Contains the name of the operating system on which NATURAL is running.
*OSVERS	Contains the version number of the operating system on which NATURAL is running.
*ROWCOUNT	Contains the number of rows deleted, updated or inserted by the last NATURAL SQL statement (searched DELETE, searched UPDATE, or INSERT with *select-expression* respectively).
*UI	Indicates the type of user interface being used: character-oriented or graphical (GUI).
*WINMGR	Contains, for graphical user interfaces, the name of the window manager being used.
*WINMGRVERS	Contains, for graphical user interfaces, the version of the window manager being used.

New Date System Variables

The following new date system variables are available: ***DAT4D**, ***DAT4E**, ***DAT4I**, ***DAT4J** and ***DAT4U**, all of which provide the year information as 4 digits. Otherwise their date representation corresponds to that of the date system variables *DATD, *DATE, *DATI, *DATJ and *DATU respectively.

Enhanced System Variable

*TPSYS

Under TIAM, the system variable *TPSYS now contains the value "TIAM" instead of "RTIO".

New System Function

SORTKEY

Several national languages contain characters (or combinations of characters) which are not sorted in the correct alphabetical order by a sort program or database system, because the sequence of the characters in the character set used by the computer does not always correspond to the alphabetical order of the characters.

For example, the Spanish letter "CH" would be treated by a sort program or database system as two separate letters and sorted between "CG" and "CI" — although in the Spanish alphabet it is in fact a letter in its own right and belongs between "C" and "D".

Or it may be that, contrary to your requirements, lower-case and upper-case letters are not treated equally in a sort sequence, that letters are sorted after numbers (although you may wish them to be sorted before numbers), or that special characters (for example, hyphens in double names) lead to an undesired sort sequence.

In such cases, you can use the new system function SORTKEY(*character-string*) to convert "incorrectly sorted" characters into other characters that are "correctly sorted" alphabetically by the sort program or database system. The values computed by SORTKEY would then only be used as sort criterion, while the original values would be used for the interaction with the end-user.

When you specify the SORTKEY function in a NATURAL program, the user exit NATUSK*nn* will be invoked (*nn* being the current language code as in the system variable *LANGUAGE). The *character-string* specified with SORTKEY will be passed to the user exit. The user exit has to be programmed so that it converts "incorrectly sorted" characters in this string into corresponding "correctly sorted" characters. The converted character string is then used in the NATURAL program for further processing.

The user exit is described in the *NATURAL Installation and Operations Manual for Mainframes*.

For further information on the SORTKEY function, see the *NATURAL Reference Manual*.

Enhanced Statements

CALL, CALLNAT and FETCH

The name of the program/subprogram to be invoked can be specified as an alphanumeric variable. With Version 2.2, the length of this variable has to be exactly 8. With Version 2.3, its length can be from 1 to 8.

CALLNAT and PERFORM
New Option AD=A

With Version 2.2, you can mark a parameter to be passed to a subprogram with AD=O (non-modifiable) or AD=M (modifiable).

With Version 2.3, you can also mark a CALLNAT parameter with AD=A (input-only): For remote subprograms executed via NATURAL remote procedure call (RPC) in a client/server environment, such a parameter will not be passed *to* the subprogram, but receive a value *from* the subprogram (see also page 91). If a subprogram is executed locally, the AD=A field will be reset to empty before the subprogram is invoked.

For subroutines, AD=A is also possible: A PERFORM parameter marked with AD=A will be reset before the subroutine is invoked and can be used to receive a value *from* the subroutine.

Internal Handling of AD=O

With Version 2.3, the internal handling of AD=O has changed. A CALLNAT/PERFORM parameter marked with AD=O is no longer passed to the subprogram/subroutine "by reference" (that is, via its address) but "by value".

For details on the passing of parameters, see the CALLNAT and PERFORM descriptions in the *NATURAL Statements Manual*.

COMPRESS

The COMPRESS statement provides the following new options:

FULL	With this option, the values of the source operands in their actual lengths — that is, including leading zeros and trailing blanks — are transferred to the target field.
	Without this option, leading zeros and trailing blanks are suppressed before the values are transferred.
NUMERIC	With this option, decimal points and signs in numeric source values are also transferred to the target field.
	Without this option, decimal points and signs are suppressed before the values are transferred.
ALL	This option can be used in conjunction with the option WITH DELIMITER(S):
	Without ALL, a delimiter is placed in the target field only between values actually transferred.
	With ALL, a delimiter is placed into the target field also for each blank value that is not actually transferred. This means that the number of delimiters in the target field corresponds to the number of source fields minus 1. This may be useful, for example, if the content of the target field is to be separated again with a subsequent SEPARATE statement.
SUBSTRING	This option, which has already been available in several other statements, is now also available in the COMPRESS statement for both the source fields and the target field. It allows you to transfer only parts of source fields and/or transfer them into a specific part of the target field.

Example of FULL Option:

```
1.   COMPRESS 'ABC '  001 INTO #TARGET WITH DELIMITER '*'

     Content of #TARGET is: ABC*1


2.   COMPRESS FULL 'ABC '  001 INTO #TARGET WITH DELIMITER '*'

     Content of #TARGET is: ABC *001
```

Example of NUMERIC Option:

```
1.   COMPRESS -123  1.23 INTO #TARGET WITH DELIMITER '*'

     Content of #TARGET is: 123*123


2.   COMPRESS NUMERIC -123  1.23 INTO #TARGET WITH DELIMITER '*'

     Content of #TARGET is: -123*1.23
```

Example of ALL Option:

```
1.   COMPRESS 'A' ' ' 'C' ' ' INTO #TARGET WITH DELIMITER '*'

     Content of #TARGET is: A*C


2.   COMPRESS 'A' ' ' 'C' ' ' INTO #TARGET WITH ALL DELIMITERS '*'

     Content of #TARGET is: A**C*
```

COMPRESS and MOVE

In order to support languages whose writing direction is from right to left, you can specify the option PM=I in the statements COMPRESS and MOVE so as to transfer the value of a source operand in inverse (right-to-left) direction.

Example 1:

```
MOVE 'XYZ' TO #A
MOVE #A (PM=I) TO #B
```

Content of #B is: ZYX

Example 2:

```
MOVE 'XYZ' TO #A
COMPRESS #A (PM=I) 'ABC' INTO #B
```

Content of #B is: ZYX ABC

DEFINE DATA

The DEFINE DATA statement provides the following new options:

- DEFINE DATA CONTEXT,
- BY VALUE and BY VALUE RESULT (in DEFINE DATA PARAMETER).

Moreover, the handling of the following is different:

- decimal digits of constant values,
- redefinition of database arrays.

DEFINE DATA CONTEXT

DEFINE DATA CONTEXT is used in conjunction with NATURAL remote procedure call (RPC). It is used to define variables that are to be available to multiple remote subprograms within one conversation, without having to explicitly pass the variables as parameters with the corresponding CALLNAT statements. See also page 89.

Only level-1 variables and redefinitions can be specified within DEFINE DATA CONTEXT; group and view definitions, however, are not possible. The variables can also be defined in a separate data area, that is, DEFINE DATA CONTEXT USING *local/parameter-data-area* is also possible.

BY VALUE and BY VALUE RESULT (in DEFINE DATA PARAMETER)

With Version 2.2, parameters are passed to a subprogram/subroutine via their addresses (that is, by reference); therefore the format/length of a field specified as parameter in a CALLNAT/ PERFORM statement have to be the same as the format/length of the corresponding field in the invoked subprogram/subroutine.

Version 2.3 provides the new option "BY VALUE" in the DEFINE DATA PARAMETER statement: With this option, you can pass parameters to a subprogram/subroutine *by value*; that is, the actual parameter values (instead of their addresses) are passed to the subprogram/subroutine. Consequently, the fields in the subprogram/subroutine need not have the same format/length as the CALLNAT/PERFORM parameters.

With this new option, you can, for example, increase the length of a field in a subprogram/subroutine (if this should become necessary due to an enhancement of the subprogram/subroutine) without your having to adjust any of the objects that invoke the subprogram/subroutine.

Example:

```
* Program
DEFINE DATA LOCAL
  1 #FIELDA (P5)
  ...
END-DEFINE
...
CALLNAT 'SUBR01' #FIELDA
...


* Subroutine SUBR01
DEFINE DATA PARAMETER
  1 #FIELDB (P9) BY VALUE
END-DEFINE
...
```

While the "BY VALUE" option applies to parameters being passed *to* a subprogram/subroutine, the new option "BY VALUE RESULT" causes parameters to be passed *by value* in both directions; that is, the actual parameter values are passed from the invoking object to the subprogram/subroutine and, on return to the invoking object, the actual parameter values are passed from the subprogram/subroutine back to the invoking object.

Decimal Digits of Constant Values

If the constant value specified after CONSTANT or INIT has more digits after the decimal point than the corresponding field, this does not lead to an error with Version 2.2. With Version 2.3, such inconsistency leads to error NAT0094 at compilation.

Example:

```
DEFINE DATA LOCAL
1 #FIELD (N2) INIT <12.25> /* no longer possible with Version 2.3
END-DEFINE
```

A compilation option (see page 34) is provided to allow you to temporarily continue to use the old Version 2.2 syntax.

Redefinition of Database Arrays

To preclude referencing errors, it is no longer possible to specify a variable index range in the redefinition of a periodic-group field or multiple-value field.

Example:

```
DEFINE DATA LOCAL
1 MYVIEW VIEW OF EMPLOYEES
  2 SALARY (I:I+2)
  2 REDEFINE SALARY
    3 MYSALARY (P9/I:I+2) /* no longer possible with Version 2.3
END-DEFINE
```

With Version 2.3, the above redefinition has to be changed to:

```
    3 MYSALARY (P9/1:3)
```

A compilation option (see page 34) is provided to allow you to temporarily continue to use the old Version 2.2 syntax.

DEFINE PRINTER

The following additional options are available for overwriting the definitions specified with the NTPRINT macro:

- **CLASS** — spool class (1 character)

- **PRTY** — listing priority (1 to 255).

DEFINE WINDOW

The minimum possible size of a window is 2 lines by 10 columns without frame, and 4 lines by 12 columns with frame. The size of a window can be specified in the SIZE clause of the DEFINE WINDOW statement.

With Version 2.2, a window size smaller than the minimum can be specified: at runtime, the size of the window is then automatically set to the minimum possible number of lines and/or columns.

With Version 2.3, the specification of too small a window size leads to error NAT1167 at compilation.

Example:

```
DEFINE WINDOW XYZ
  SIZE 4 * 8  FRAMED OFF   /* Version 2.2: Size set to 4 * 10 at runtime.
                           /* Version 2.3: Syntax error.
```

A compilation option (see page 34) is provided to allow you to temporarily continue to use the old Version 2.2 syntax.

DISPLAY

If there are multiple DISPLAY statements in a program, the first DISPLAY statement determines the column headers to be used.

With Version 2.2, this is not always handled correctly in the case of the first DISPLAY statement being a DISPLAY VERTICALLY statement without AS clause: Without AS clause, no column headers should be displayed; instead, however, the column headers from the next DISPLAY statement are displayed.

With Version 2.3, this faulty behaviour has been corrected.

DIVIDE

With Version 2.3, DIVIDE statements using both the GIVING and the REMAINDER option may in some cases — if the dividend (operand2) has more decimal positions than the result field — give different results for the REMAINDER field. However, these results will be of a greater precision.

Note: This change will only affect programs that are newly compiled under Version 2.3. Programs compiled under Version 2.2 and executed under Version 2.3 will not be affected.

Example:

```
DEFINE DATA LOCAL
  1 #RESULT   (N2)
  1 #REMAIN   (N4.1)
END-DEFINE
*
DIVIDE 3 INTO 10.5 GIVING #RESULT REMAINDER #REMAIN
*
**                      #RESULT  #REMAIN
** VALUES WITH VERSION 2.2:  3       0.0
** VALUES WITH VERSION 2.3:  3       1.5
**
END
```

EJECT

To enhance the clarity of programs and avoid possible ambiguities in the source code, the keyword LESS in Syntax 2 of the EJECT statement is no longer optional, but required.

With Version 2.2, the shortest possible form is:

```
EJECT operand1
```

With Version 2.3, it is:

```
EJECT LESS operand1
```

ESCAPE

With Version 2.2, an ESCAPE TOP or ESCAPE BOTTOM statement within an ON ERROR statement block leads to an error at runtime. With Version 2.3, this invalid coding is already intercepted at compilation.

Moreover, it is no longer allowed to place an ESCAPE TOP statement within an AT START OF DATA statement block.

FIND

The comparison logic for multiple-value fields in the WITH clause of the FIND statement has been changed so as to be in line with the comparison logic in other statements (e.g. IF).

Four different forms of the FIND statement can be distinguished (the field MU in the following examples is assumed to be a multiple-value field):

1.   `FIND XYZ-VIEW WITH` **`MU = 'A'`**

With Versions 2.2 and 2.3, this statement returns records in which *at least one occurrence* of MU has the value "A".

2.   `FIND XYZ-VIEW WITH` **`MU NOT EQUAL 'A'`**

With Version 2.2, this statement returns records in which *none of the occurrences* of MU has the value "A" (same as **4.**).
With Version 2.3, this statement returns records in which *at least one occurrence* of MU does *not* have the value "A".

3.   `FIND XYZ-VIEW WITH` **`NOT MU NOT EQUAL 'A'`**

With Version 2.2, this statement returns records in which *at least one occurrence* of MU has the value "A" (same as **1.**).
With Version 2.3, this statement returns records in which *every occurrence* of MU has the value "A".

4.   `FIND XYZ-VIEW WITH` **`NOT MU = 'A'`**

With Versions 2.2 and 2.3, this statement returns records in which *none of the occurrences* of MU has the value "A".

This means that if you newly compile under Version 2.3 existing Version 2.2 programs containing FIND statements of the forms **2.** and **3.**, they will return different results.

A compilation option (see page 34) is provided which allows you to detect such statements. If you activate this option, error NAT0998 will be returned for every FIND statement of form **2.** or **3.**   detected at compilation.

Should you in these cases wish to continue to get the same results as with Version 2.2, you have to change the statements as follows:

2.   `FIND XYZ-VIEW WITH` **`MU NOT EQUAL 'A'`**   `change to:`   **`NOT MU = 'A'`**

3.   `FIND XYZ-VIEW WITH` **`NOT MU NOT EQUAL 'A'`** `change to:`   **`MU = 'A'`**

FIND, GET, HISTOGRAM, READ and STORE

With Version 2.2, it is possible in a non-NATURAL SECURITY environment in reporting mode to specify an ADABAS file number as *view-name*.

With Version 2.3, it is no longer possible to specify an ADABAS file number as *view-name*. This will lead to a compilation error (NAT0980).

FIND and READ

The statements FIND and READ provide a new option "STARTING WITH ISN = *operand*". This option may be used for repositioning within a FIND/READ loop whose processing has been interrupted, to easily determine the next record with which processing it to continue. This is particularly useful if the next record cannot be identified uniquely by any of its descriptor values. It can also be useful in a distributed client/server application where the reading of the records is performed by a server program while further processing of the records is performed by a client program, and the records are not processed all in one go, but in batches.

This option is provided for access to ADABAS databases and, with some restrictions, also for VSAM and DL/I.

HISTOGRAM and READ

The statements HISTOGRAM and READ provide new options for reading records in descending sequence in order to support the "read backwards" feature of ADABAS, VSAM and SQL databases.

Two options are possible:

- Static backward reading — by specifying the keyword DESCENDING.

- Variable forward/backward reading — by specifying the keyword VARIABLE followed by a variable (format/length A1) which determines the reading direction. The variable can contain the value "A" (for "ascending") or "D" (for "descending"). This allows you to determine the reading sequence at runtime when the HISTOGRAM or READ loop starts.

Notes:
The VARIABLE option is not possible for SQL databases.

The default sequence is ascending. The new keyword ASCENDING is provided to allow you to explicitly specify ascending sequence.

Note for ADABAS:
For READ statements, the "read backwards" feature requires ADABAS Version 6.1 or above; for Version 6.1, the ZAPs distributed with early warnings ADA612–007 and ADA613–002 respectively have to be applied.
For HISTOGRAM statements, the "read backwards" feature requires ADABAS Version 6.2 or above.

Example of DESCENDING Option:

```
READ EMPLOYEES IN DESCENDING SEQUENCE BY NAME = 'SMITH'
```

This statement returns all names in descending sequence, starting with the name "SMITH".

Example of VARIABLE Option:

```
DEFINE DATA LOCAL
1 #DIRECTION (A1) INIT <'A'>   /* 'A' = ASCENDING
1 #EMPVIEW VIEW OF EMPLOYEES
   2 NAME
   ...
END-DEFINE
...
IF *PF-KEY = 'PF7'
  THEN MOVE 'D' TO #DIRECTION
END-IF
READ #EMPVIEW IN VARIABLE #DIRECTION SEQUENCE BY NAME = 'SMITH'
  ...
END-READ
...
```

MOVE

With Version 2.2, if the value to be moved with MOVE RIGHT JUSTIFIED is longer than the target field, the value will be truncated on the *right-hand* side before being placed into the target field.

With Version 2.3, if the value to be moved with MOVE RIGHT JUSTIFIED is longer than the target field, the value (after the removal of trailing blanks) will be truncated on the *left-hand* side before being placed into the target field.

Example:

```
DEFINE DATA LOCAL
  1 #SOURCE (A6) INIT <' ABC  ' /* 1 leading blank and 2 trailing blanks
  1 #TARGET (A3)
END-DEFINE
*
MOVE #SOURCE TO #TARGET
*
** CONTENTS OF #TARGET – WITH VERSION 2.2: ' AB'
**                     – WITH VERSION 2.3: 'ABC'
**
END
```

Note: For the new handling to apply to programs compiled with the NATURAL OPTIMIZER COMPILER, the programs have to be newly compiled; otherwise the old handling will continue to apply.

NEWPAGE

To enhance the clarity of programs and avoid possible ambiguities in the source code, the keywords TOP and LESS of the NEWPAGE statement are no longer optional, but required.

With Version 2.2, the shortest possible forms are:

```
NEWPAGE EVEN
NEWPAGE operand1
```

With Version 2.3, they are:

```
NEWPAGE EVEN TOP
NEWPAGE LESS operand1
```

OPTIONS

With Version 2.2, this statement can be used to control the NATURAL OPTIMIZER COMPILER (MCG parameter).

With Version 2.3, it is possible to specify with the OPTIONS statement all compilation options that can be set with the new system command NOCOPT (see page 36).

PRINT

It is no longer possible to specify the LS parameter with the PRINT statement (as it has no effect anyway). With Version 2.2, this does not lead to an error; with Version 2.3, it will lead to error NAT0934.

READ

With the new option WITH REPOSITION (which can only be applied to VSAM databases), you can reposition to another start value for the database records read within the active READ loop. Processing of the READ statement then continues with the new start value.

The repositioning is triggered by the value of the system variable *COUNTER being reset to "0"; that is, the new start value is used as soon as *COUNTER is "0".

Example:

```
DEFINE DATA LOCAL
1 MYVIEW VIEW OF ...
  2 NAME
1 #STARTVAL (A20) INIT <'A'>
1 #ATTR    (C)
END-DEFINE
...
SET KEY PF3
...
READ MYVIEW WITH REPOSITION BY NAME = #STARTVAL
  INPUT (IP=OFF AD=O) 'NAME:' NAME /
    'Enter new start value for repositioning:' #STARTVAL (AD=MT CV=#ATTR) /
    'Press PF3 to stop'
  IF *PF-KEY = 'PF3'
    THEN STOP
  END-IF
  IF #ATTR MODIFIED
    THEN RESET *COUNTER
  END-IF
END-READ
...
```

MASK Option (Logical Condition)

With the MASK option, you can specify a new character "/" (slash) to check if a value ends with a specific character or string of characters.

Example 1:

```
IF #FIELD = MASK (*'E'/)
```

This condition will be true if there is either an "E" in the last position of the field, or the last "E" in the field is followed by nothing but blanks.

SUBSTRING Option (Various Statements)

With Version 2.2, invalid or inconsistent values for the starting position and/or the length of the field portion in a SUBSTRING option lead to errors at runtime.

With Version 2.3, such invalid/inconsistent values in a SUBSTRING option are already intercepted at compilation (error NAT0471).

Incomplete Statement Blocks (Various Statements)

With Version 2.2, an empty statement block (for example, a FOR or REPEAT processing loop that does not contain any statements) may in some cases not lead to a compilation error.

With Version 2.3, this inconsistency has been corrected, and any empty — that is, syntactically incomplete — statement block will lead to an error at compilation. If you wish a statement block to intentionally perform no function, insert an IGNORE statement.

Assignment of Numeric Value to Alphanumeric Field (Various Statements)

With Version 2.2, when a numeric value is assigned to an alphanumeric field, any leading bytes containing hexadecimally H'x0' will be truncated.

With Version 2.3, when a numeric value is assigned to an alphanumeric field, only leading bytes containing hexadecimally H'F0' will be truncated, whereas bytes containing any other H'x0' value will be retained. This behaviour is consistent with the handling of such values in output statements (e.g. WRITE).

Database Field Names (any Database Statement)

With Version 2.2, two-character database field names in a database statement are interpreted as field short names (as used by the underlying database system), whereas other database field names are interpreted as field long names (as defined for NATURAL in the corresponding DDM). For some database systems, this may lead to long names erroneously interpreted as short names.

With Version 2.3, it is possible to set an option with the new system command COMPOPT (see page 34) so that database field names will always be interpreted as long names, regardless of their length. This will avoid possible misinterpretations of database field names in programs.

No Upper-Case Translation for System Variable *COM

As documented in the *NATURAL 2.2 Reference Manual*, it is intentional that no translation to upper case is performed for any input entered into a *COM field.

With Version 2.2, however, the explicit specification of the field attribute "AD=T" (translation to upper case) for a *COM field, although ignored at runtime, is not rejected at compilation.

With Version 2.3, the specification of AD=T for a *COM field will lead to a syntax error (NAT0335).

Unique Statement Labels

To preclude situations in which the wrong field might be referenced because of a possible ambiguity between a statement label and a field qualifier, as of Version 2.3.1 a statement label (not counting the period) must not have the same name as a variable defined at level 1.

Note: On other platforms, this restriction was already implemented with Version 2.2.

SQL

Flexible SQL

NATURAL's flexible SQL has been enhanced to support "dynamic SQL": The flexible SQL can contain a variable which contains SQL text; at runtime, the variable will then be replaced by the SQL text.

SYSTEM COMMANDS AND UTILITIES

This chapter contains information on:

- new system commands,

- enhanced system commands,

- removed system commands,

- SYSPARM — a new utility for dynamic parameters,

- the NATUNLD and NATLOAD utilities,

- the SYSBPM utility,

- the SYSDBA utility,

- the SYSDDM utility,

- the SYSMAIN utility,

- the SYSNCP utility,

- the SYSRDC utility,

- the SYSTP utility,

- the SYSTRANS utility,

- the recording utility,

- the debugging utility,

- the call statistics utility,

- editors.

New System Commands

COMPOPT — Compilation Options

This new system command allows you to set various options which will affect the way in which a NATURAL object is compiled. The following options will be available:

- **Keyword Checking**
  (corresponds to the Version 2.2 profile parameter KC, which has been be removed).

- **Parameter Checking for CALLNAT Statements**
  This new option allows you to check if the number of parameters specified in a CALLNAT statement correspond with those in the subprogram to be invoked — provided that subprogram already exists (if the subprogram does not exist, this option has no effect).

- **Database Short Field Names**
  This new option determines how database field names are interpreted in NATURAL programs. Two options are possible:
  - Database field names are considered long names (as defined in the corresponding DDM) — except 2-character field names, which are considered short names (as used by the underlying database system). This is the default.
  - All database field names are considered long names, regardless of their length. This avoids possible misinterpretations of database field names in programs.

- **Internal Representation of Positive Sign of Packed Numbers**
  This new option determines whether the positive sign of packed numbers is represented internally as H'C' or as H'F' (default is H'F'); see also page 99.

- **Global Format IDs**
  This new option allows you to control NATURAL's generation of global format IDs so as to influence ADABAS's performance for format buffer translations.

- **Detect Inconsistent Comparison Logic in FIND Statements**
  This option may be used to search for FIND statements whose WITH clauses use multiple-value fields in a way that is no longer consistent with the enhanced Version 2.3 comparison logic (see page 24).

- **Applicability of TS Profile Parameter**
  This option may be used to determine whether the profile parameter TS (translate output for locations with non-standard lower-case usage) is to apply.

- **Compatibility Option — Allow Old Version 2.2 Syntax** (see below).

It is also possible to set the COMPOPT options with the new macro NTCMPO in the NATURAL parameter module, and dynamically with the corresponding new profile parameter CMPO.

Compatibility Option — Allow Old Version 2.2 Syntax

Some inconsistent syntax constructions that are not intercepted by Version 2.2 lead to a syntax error with Version 2.3:

- DEFINE DATA — inconsistent number of decimal digits in constant value (see page 20).

- DEFINE DATA — redefinition of database array with variable index range (see page 20).

- DEFINE WINDOW — specified window size smaller than minimum (see page 21).

To allow you a smooth transition from Version 2.2 to Version 2.3, the compatilibity option of COMPOPT can be set: the above syntax constructions will then *not* lead to a syntax error. Thus you will be able to compile your existing programs under Version 2.3 until you have adjusted them to the Version 2.3 requirements.

Note: The compatibility option will be available only for a limited period of time to allow you a smooth transition to the Version 2.3 syntax. It will be removed again with one of the next releases of NATURAL.

LASTMSG — Information on Last Error Situation

With this new system command, you can display additional information about the error situation which has occurred last.

When NATURAL displays an error message, it may in some cases be that this error is not the actual error, but an error caused by another error (which in turn may have been caused by yet another error, etc.) In such cases, the LASTMSG command allows you to trace the issued error back to the error which has originally caused the error situation.

When you enter the LASTMSG command, you will get — for the error situation that has occurred last — the error message that has been displayed, as well as all preceding (not displayed) error messages that have led to this error.

When you mark one of these messages with the cursor, you will get the following information on the corresponding error: error number; number of the line in which the error occurred; name, type and level of the object that caused the error; name, database ID and file number of the library containing the object; error class (error issued by NATURAL or by user application); error type (runtime, syntax, command execution, session termination, program termination, remote procedure call); date and time at which the error occurred.

NOCOPT — NATURAL OPTIMIZER COMPILER Options

This new system command corresponds to the Version 2.2 profile parameter MCG (which has been removed).

It allows you to set within a NATURAL session all options that are available to control the NATURAL OPTIMIZER COMPILER. For details on these options, see the *NATURAL OPTIMIZER COMPILER Manual*.

Enhanced System Commands

CATALL

The CATALL command provides the following enhancements:

- Performance of CATALL processing has been improved considerably, particularly for large libraries.

- With a new selection function, you can select from a list those objects you wish to be processed by CATALL. Optionally, you can store the selection information and re-use it in another CATALL command to process the same objects again.

- With asterisk notation (*) and wildcard notation (?), you can specify a range of object names (in the same manner as with the system command LIST); or you can determine a range of object names by specifying a start value and an end value.

- With Version 2.2, CATALL can be used to SAVE, CATALOG and STOW objects. With Version 2.3, it can also be used to CHECK objects.

- With Version 2.2, some CATALL processing options are only available via a direct command option. With Version 2.3, all CATALL options can be selected on the CATALL screen.

- The direct command syntax of CATALL has been enhanced.

DELETE, PURGE, SCRATCH and UNCATALOG

With Version 2.2, the syntax of the DELETE command allows the specification of object types, object names and source/object/both indicators in combinations which in some cases might possibly be misinterpreted.

With Version 2.3, the syntax of the DELETE command has been revised and enhanced to preclude such cases. In particular, the object type specification has to be preceded by the keyword "TYPE"; "S", "O" and "B" have to be written out as "SOURCE", "OBJECT" and "BOTH" respectively, and they have to be specified *before* the object name. For the full new syntax, see the *NATURAL User's Guide for Mainframes*. As the new syntax is not fully compatible with the old syntax, you may have to adjust existing DELETE commands that appear in fixed form (for example, in batch jobs or STACK statements).

To make the commands DELETE, PURGE, SCRATCH and UNCATALOG more uniform, the characters to be used to mark an object on a selection list are now the same for all four commands: "S", "O" or "B" (= delete source, object, or both).
(For PURGE, SCRATCH and UNCATALOG, only the character corresponding to the command's functions is available, of course.)

GLOBALS

The GLOBALS command displays a screen with all session parameters that can be modified with the GLOBALS command. With Version 2.2, only a subset of these parameters is displayed.

The function-key settings are no longer displayed on the GLOBALS screen; they are displayed via the KEY command instead (see below).

KEY

It is possible to enter the KEY command without any parameters: it then displays a screen with the current function-key setting. On this screen, you can modify individual settings.

LIST

The LIST command provides the following enhancements:

- To list a certain range of objects, you can use the new notations ">" and "<" in the object name to specify a start value or end value respectively for the list of objects.

- When a list of object is displayed, there are fields immediately underneath the column headings showing the selection criteria for the current list. You can change the selection criteria by overwriting the values of these fields. This also allows you to use date, version, etc. as selection criteria.

- It is possible to scroll directly to the end of the list of objects. Moreover, the overall performance for backward scrolling and positioning has been improved.

- The list of objects can now be printed. The page size for printing (for a list of objects as well as for an individual object) can be adjusted as required.

- The LIST subcommands that are available for a listed source are also available in batch mode.

- With Version 2.2, the EXPAND option of the LIST command only applies to the current library. With Version 2.3, it applies to the current library **and** its steplibs.

- Several new function codes are available for marking an object on the selection list for a function to be performed (for example, CA for CATALOG, LF for LIST FORMATTED).

- The display of a list of DDMs (LIST VIEW) has been adapted to coincide with the display of the other objects.

- The information displayed by the LIST DIR command also includes the setting of the profile parameter MCG (NATURAL OPTIMIZER COMPILER).

SCAN

The SCAN command provides the following enhancements:

- A new NOT EQUAL option allows you to ascertain in how many objects the search string is **not** contained.

- After the search, instead of immediately displaying the source-code lines, you can first display the number of objects in which the search string has been found.

TECH

The TECH command has been enhanced to provide additional error information.

XREF

Apart from the existing values, you can specify the new value "DOC". XREF DOC corresponds to XREF FORCE, except that no cross-reference data will be generated.

Removed System Commands

ADHOC and CREATE

As of Version 2.3, the system commands ADHOC and CREATE are no longer available.

The functionality provided by these commands (line-by-line creation of very short programs) is in fact a relic from Version 1.2 of NATURAL, and both commands are little used these days — what with the full-screen NATURAL program editor with its full range of editing functions at your disposal.

Note: Should you still need to do single-line editing in batch mode, the system command EDT will continue to be available to you.

SYSPARM — New Utility for Dynamic Parameters

The new utility SYSPARM allows to invoke NATURAL with dynamic profile parameters in a more comfortable way.

With Version 2.2, when you invoke NATURAL with dynamic profile parameters, you have to specify a whole string of individual parameters — each time you invoke NATURAL.

With Version 2.3, you can specify a string of profile parameters once, store this string under a profile name, and then invoke NATURAL with only one dynamic parameter: PROFILE=*profile-name*. The string of parameters stored under that profile name will then be passed to NATURAL as dynamic parameters.

You create and maintain these profiles (i.e. strings of parameters) with the new utility SYSPARM.

The new dynamic parameter PROFILE provides the following special options:

- **PROFILE=AUTO** — NATURAL will take the current TP user ID (as contained in the system variable *INIT–USER) as profile name, which means that the profile defined under this ID will be used.
  If no profile is defined under that ID, a profile named "AUTO" will be used instead (if defined); you can define such an "AUTO" profile as default profile for users without individual profiles.

- **PROFILE=TERMINAL** — NATURAL will take the current terminal ID (as contained in the system variable *INIT–ID) as profile name, which means that the profile defined under this ID will be used.

- **PROFILE=PROGRAM** — NATURAL will take the name of the program currently executing as NATURAL (as contained in the system variable *INIT–PROGRAM) as profile name, which means that the profile defined under this name will be used.

The dynamic parameter PROFILE is evaluated before any other dynamic parameters are evaluated, which means that individual parameters in the profile can be overridden by dynamic parameters specified in addition to the PROFILE parameter.

Note: With Version 2.2, the SYSPARM utility and PROFILE parameter have already been available in a preliminary form for testing purposes.

NATUNLD and NATLOAD Utilities

Library SYSUNLD

The utilites NATUNLD and NATLOAD are no longer contained in the library SYSTEM, but in a new library of their own, SYSUNLD.

DBID/FNR

If no DBID/FNR for the source/target library is specified with NATUNLD/NATLOAD, the DBID/FNR definitions from the corresponding NATURAL SECURITY library profile will be used.

Unloading of Stowed Objects

The fact that a saved object and a cataloged object have the same name does not guarantee that they actually belong together. Therefore NATUNLD provides are new function which allows you to unload only objects whose saved and cataloged forms were stored at the same time. The check that the saving and cataloging of an object occurred at the same time ensures that both forms of an object belong together.

Unloading Without Symbol Tables

NATUNLD provides a new option to unload objects without their corresponding symbol tables. This will reduce the amount of disk storage required. However, this will only be useful for a production environment, as several application development functions which require the symbol tables will then not be available; moreover, the profile parameter RECAT=ON will not apply.

Different Target Library

With Version 2.2, the objects are loaded into the target library as specified with the NATUNLD utility when the objects are unloaded.

With Version 2.3, NATLOAD allows you to specify a different target library.

Replace Only Old Objects in Target Library

With Version 2.2, the Replace option of the NATLOAD utility allows you to overwrite either all or no existing objects in the target library.

With Version 2.3, a new option allows you to overwrite existing objects depending on their timestamp: only older objects (that is, objects which were saved/cataloged before the objects of the same names to be loaded) will be overwritten.

Delete Instructions

With NATUNLD, you can write delete instructions for specific objects to the work file. When these instructions are loaded with NATLOAD from the work file into the target environment, they cause the specified objects to be deleted *from the target environment*.

User Exits with Different Source Names

The avoid the overwriting of modified user exit sources by update installations, the names of the sources of the user exits provided by NATUNLD/NATLOAD are different from the names of the corresponding user exit objects actually invoked. The user exits are provided in the new library SYSUNLD. See the *NATURAL Utilities Manual for Mainframes* for details.

Old NATLOAD Cannot Load New Work Files

Work files that were created with NATUNLD Version 2.3 cannot be loaded with NATLOAD Version 2.2.

Version 2.1 Batch Parameter Syntax No Longer Supported

The old Version 2.1 ULDMAIN/INPL syntax for supplying parameters in batch mode is no longer supported by NATUNLD/NATLOAD.

Direct Command for Unloading DDMs

With Version 2.2, the following syntax of the direct command to unload DDMs is accepted and processed as valid — although it is incorrect:

NATUNLD VIEW *view-name* **WHERE DBID** *nn* **FNR** *nn*

With Version 2.3, this inconsistency has been corrected, and the above syntax will lead to an error.

The correct syntax (as documented in the *NATURAL 2.2.8 Utilities Manual for Mainframes*) is:

NATUNLD VIEW *view-name* **WHERE DIC (***nn,nn***)**

SYSBPM Utility

Blacklist Maintenance

For easier maintenance of blacklist entries, you can place a set of blacklist entries in a NATURAL object; in the blacklist itself, you then only have to specify the name of that object (instead of the individual blacklist entries).

Pre-Load List

You can define a list of objects that will automatically be loaded into the NATURAL buffer pool and marked as resident as soon as the buffer pool is initialized.

It is possible to define a different pre-load list for each buffer pool.

SYSDBA Utility

The SYSDBA utility is no longer available. The functions it contained have been transferred to the following locations:

- **SYSPROD** (Display Product Information), **SYSPROF** (Display System File Information) and **ROUTINES** (Display Subroutines Used) are now available as system commands (as described in the *NATURAL User's Guide for Mainframes*).

- **BUS** (Buffer Usage Statistics) and **SYSFILE** (Work File and Printer Availability) are now part of the utility SYSTP (as described in the *NATURAL Utilities Manual for Mainframes*).

- **ADACALL** (Issue ADABAS Calls) is now contained in the new library SYSADA.

- **ULDOBJ** (Link NATURAL Cataloged Objects to the Nucleus) is now contained in the new library SYSMISC.

- **DISPLAY** (Display Directory Information) and **RECOVER** (Review Editwork Member) are no longer available. The functionality provided by these functions is also provided by the SYSMAIN utility (it has already been provided by SYSMAIN with Version 2.2).

SYSDDM Utility

Generation of Null-Value Indicator Fields

The SYSDDM utility has been enhanced to support the generation of null-value indicator fields for ADABAS files (see page 69).

Revised User Interface

The user interface of the SYSDDM utility has been slightly revised to be more consistent and user-friendly.

In particular, the "Read DDM" function has been integrated into the "Edit DDM" function.

SYSMAIN Utility

Multiple Commands in Batch Mode

In batch mode, you can now enter multiple SYSMAIN commands without having to newly log on to SYSMAIN again.

New Direct Command LISTLIB (for Batch Mode only)

The new direct command LISTLIB is provided for batch mode to list only library names.

User Exits

Three new user exits are available:

- MAINEX08 can be used to determine further processing if no objects are found for a command in batch mode.

- MAINEX09 can be used to determine the action to be taken in case of a termination error in batch mode.

- MAINEX10 can be used to determine the action to be taken in case of an invalid command in batch mode.

The parameters for the user exits MAINEX01, MAINEX02, MAINEX04 and MAINEX06 have been changed.

The names of the user exits' sources and objects are now different to ensure that the overwriting of the sources by an update installation does not affect the objects. The source codes of the subprograms are stored under the names SM–UX–*nn* (*nn* = 01 to 10) in library SYSMAIN. To make a user exit available, you have to stow the corresponding source under the name MAINEX*nn* (either in the library SYSMAIN or in one of its steplibs).

SYSNCP Utility

New Direct Command QUICK–EDIT

The new direct command QUICK–EDIT allows you to quickly define local/global functions, as well as the corresponding runtime actions, by entering keywords or IKNs directly. This may be helpful for extremely large command processors. Note, however, that the location is not verified.

New Editor Commands in Function Editor

Several new editor commands are available in the function editor; see *NATURAL Utilities Manual for Mainframes* for details.

New User Exit NCP–REDM

The new user exit NCP–REDM allows you to define default values for runtime action definitions.

DDM "COMMAND"

The DDM "COMMAND" has been changed: the PRIVATE–... fields have been removed. If these fields are still used in an application, this will lead to a syntax error.

Obsolete Options

The following options, which had become obsolete, have been removed:
- the options "Private Sequence Allowed" and "Private Synonyms Allowed" in Processor Header Maintenance 1 (Keyword Runtime Options);
- the option "Private Prefetch" in Processor Header Maintenance 3 (Miscellaneous Options);
- the option "NCP Logo Screen" in Session Profile Maintenance, Part 3.

User Interface

The "private synonyms" functionality had been removed with NATURAL Version 2.2.8; however, "private synonyms" still appeared on some Administrator Services screens. This inconsistency has been rectified.

SYSRDC Utility

Additional Data-Collecting Events

In addition to the existing events, the SYSRDC utility allows you to collect monitoring data at the following point (events) within NATURAL:

- at session initialization and termination,

- before a terminal I/O,

- before a non-NATURAL program is called,

- after a non-NATURAL program is called,

- at the execution of a statement.

Activation via Profile Parameter

With Version 2.3, you can activate/deactivate SYSRDC simply by setting the new profile parameter RDCSIZE. (The profile parameter USERBUF is no longer used for this purpose.)

SYSTP Utility

Monitor Function under all TP Monitors

With Version 2.2, the Monitor function has been available under UTM only. With Version 2.3, it is available for NATURAL under all supported TP monitors.

The Monitor function provides you with statistical information about screen transactions in NATURAL sessions. This information may be used to track down possible reasons for unsatisfactory performance.

With the Monitor function, you can display terminal-related statistics and program-related statistics, either for an individual terminal/program or for all active terminals/programs. The statistical information includes, for example:

- the number of screen I/Os,

- the amount of data transferred to/from the screen,

- the average duration of screen transactions,

- the processing time within NATURAL,

- the time waiting for response from ADABAS,

- the number of accesses to ADABAS user files and NATURAL system files.

SYSTRANS Utility

The SYSTRANS utility command provides the following enhancements:

- The use of SYSTRANS can now be controlled and restricted via NATURAL SECURITY.

- In addition to the existing objects, you can now also transfer command processors.

- Instead of making the selection of objects to be transferred by object types, you can now also transfer objects by library.

- Within SYSTRANS, you can now also invoke functions via direct commands.

- A new user exit allows you to invoke SYSTRANS functions from within your NATURAL applications.

Recording Utility

File for Recordings

With Version 2.2, recordings (that is, the data recorded by the NATURAL Recording utility) are stored in the system file FNAT.

With Version 2.3, the new profile parameter RFILE allows you to determine where recordings are to be stored: in the system file FNAT, in the system file FUSER, or in the scratch-pad file.

Debugging Utility

The debugging utility is not yet available with Version 2.3.1. It will be available with Version 2.3.2.

Call Statistics Utility

The call statistics (application testpath statistics) utility is not yet available with Version 2.3.1. It will be available with Version 2.3.2.

Editors

Editor Profiles

With Version 2.2, editor profiles can be stored in the FNAT or FUSER system file.

With Version 2.3, they can also be stored in the scratch-pad file.

PROFILE PARAMETERS AND PARAMETER MODULE

This chapter describes the changes concerning the NATURAL parameter module and its profile parameters and macros. The following information is provided:

- dynamic profile parameters,

- 2-byte database IDs and file numbers,

- new profile parameters,

- enhanced profile parameters,

- removed profile parameters,

- new macros,

- enhanced macros,

- removed macros.

Dynamic Profile Parameters

Dynamic Overriding of All Parameter Module Settings

With Version 2.3, you can specify all NATURAL profile parameters dynamically (except CSTATIC and ISIZE). This means that you will only need to assemble the NATURAL parameter module once at installation, and can then override dynamically any settings that do not suit your requirements.

Moreover, the new utility SYSPARM is available to make the invoking of NATURAL with dynamic profile parameters easier and more comfortable (see page 41).

New Dataset CMPRMIN

The NATURAL dataset CMPRMIN, which can be used to supply dynamic NATURAL profile parameters and which has been available under MVS and TSO, is also available in batch mode under BS2000 and VSE, in single-user mode under TIAM, and in the BMP environment under IMS/TM.

The parameters need no longer be specified within the JCL/JCS used to invoke NATURAL, but in this new dataset, which in turn is specified in the JCL/JCS. Thus, you need no longer modify your JCL/JCS whenever you change a profile parameter specification. Moreover, unlike in the JCL/JCS, in the CMPRMIN dataset the length of the string of dynamic parameters is not limited.

Printing of Dynamic Profile Parameters in Batch Mode

For batch-mode environments, the new profile parameter PLOG allows you to print the list of NATURAL profile parameters that were specified dynamically when NATURAL was invoked. This allows you, for example, to see at a glance which profile parameters were specified dynamically (including PROFILE= and SYS= specifications) in a CMPRMIN dataset.

2-Byte Database IDs and File Numbers

In accordance with the enhancements of ADABAS Version 6, you can now specify values above 255 for database IDs and file numbers. This applies to all profile parameters and parameter module macros in which you can specify an ADABAS database ID or file number.

New Profile Parameters

Several new profile parameters are available with Version 2.3. The following list provides an overview of the new profile parameters. Depending on their functions, more detailed information on some of these parameters can be found in the corresponding sections of these *Release Notes*.

The default values of the new profile parameters are set so as to be compatible with the behaviour of Version 2.2.

Parameter	Function
ADANAME	Specification of ADABAS link routine name (see page 103).
ADASBV	Support of ADABAS "security-by-value" feature.
BPI	Buffer pool initialization; corresponds to new NTBPI macro (see page 62).
BPLIST	Specification of buffer pool preload list (subfunction of BPI parameter).
BPNAME	Specification of global buffer pool name (subfunction of BPI parameter).
BPPROP	Propagation of global buffer pool changes (see page 75).
BPSFI	Search sequence for objects in buffer pool and database (see page 76).
BSIZE	Determine size of ENTIRE BROKER buffer.
CANCEL	Specify command for session cancellation with dump (for debugging).
CCTAB	Set up table of printer control sequences.
CM	Suppress NEXT/MORE mode (see page 101).
CMPO	Set compilation options; corresponds to new NTCMPO macro (see page 63).
CVMIN	Determines whether or not a control variable is to be set to "MODIFIED" when the value of the field to which the control variable is attached is overwritten by an *identical* value.
DB	Specify database types and options dynamically; corresponds to NTDB macro (see page 103).
DBOPEN	Issue database open despite blank ETID (see page 103).
DBROLL	Specify number of datbase call before roll-out (see page 103).

Parameter	Function
DFOUT	Determine format of date values for output; used in conjunction with year 2000 (see page 86).
DFSTACK	Determine format of date values placed on stack; used in conjunction with year 2000 (see page 86).
DSC	Switch off data stream compression for 3270-type terminals.
EDPSIZE	Determine size of auxiliary buffer pool for SOFTWARE AG Editor.
ENDBT	Issue BACKOUT TRANSACTION statement at session end (see page 104).
ENDMSG	Suppress display of session-end message (see page 101).
ESCAPE	Deactivate terminal commands "%%" and "%." (see page 101).
ETEOP	Issue END TRANSACTION statement at end of program (see page 104).
ETIO	Issue END TRANSACTION statement upon terminal I/O (see page 104).
ETPSIZE	Determine size of ENTIRE TRANSACTION PROPAGATOR buffer.
ETRACE	Activate external trace function (for debugging only).
FCDP	Suppress display of filler character for dynamically protected input fields (see page 106).
HCAM	Determine hardcopy access method to be used (see page 77).
IDSIZE	Determine size of buffer for NATURAL/IDMS interface.
ITERM	Terminate session in case of initialization error (see page 101).
ITRACE	Activate internal trace function (for debugging only).
KAPRI	Control of Kanji printing (see page 107).
MAXROLL	Specify number of CMROLL calls before roll-out (under COM-PLETE and CICS only).
MONSIZE	Determine size of buffer for SYSTP Monitor function.
MSGSF	Avoid truncation of error messages in windows (see page 102).
NAFSIZE	Determine size of NATURAL ADVANCED FACILITIES work buffer.
NAFUPF	Specify name of NATURAL ADVANCED FACILITIES user profile.
NUCNAME	Specify name of shared nucleus.

Parameter	Function
OPF	Disallow overwriting of protected fields by helproutines (see page 106).
OPT	Control of NATURAL OPTIMIZER COMPILER; corresponds to new NTOPT macro (see page 63).
PLOG	Logging of dynamic profile parameters in batch mode (see page 52).
POS22	Compatibility option for POS system function and MARK option of INPUT/REINPUT statement with NATURAL OPTIMIZER COMPILER.
PRINT	Specify print files dynamically; corresponds to new NTPRINT macro (see page 64).
RCFIND	Handling of response code 113 for FIND statements (see page 104).
RCGET	Handling of response code 113 for GET statements (see page 104).
RDCSIZE	Determine size of SYSRDC buffer.
REINP	Suppress internal REINPUT for invalid data (see page 105).
RFILE	Determine storage location for recordings (FNAT or FUSER system file or scratch-pad file).
RPC	Specify Remote Procedure Call options dynamically; corresponds to new NTRPC macro (see page 65).
SCTAB	Corresponds to new NTSCTAB macro (see page 65).
SORT	Specify sort options dynamically; corresponds to new NTSORT macro (see page 67).
SUBSID	Identifies the name of the subsystem to be used (see page 83).
TAB	Corresponds to new NTTAB macro (see page 65).
TABA1, TABA2	Correspond to new NTTABA1 and NTTABA2 macros (see page 66).
TABL	Corresponds to new NTTABL macro (see page 66).
TAB1, TAB2	Correspond to new NTTAB1 and NTTAB2 macros (see page 66).
TRACE	Define components to be traced (for debugging only).
TTYPE	Set terminal type at session initialization (see page 105).
UDSSIZE	Determine size of work buffer for NATURAL/UDS interface.

Parameter	Function
USER	Restrict use of parameter strings (as specified in SYSPARM profiles, NTSYS macros and CMPRMIN datasets).
UTAB1, UTAB2	Correspond to new NTUTAB1 and NTUTAB2 macros (see page 68).
WPSIZE	Determine size of NATURAL work pools.
YD	Set year differential (in analogy to time and day differential parameters TD and DD).
YSLW	Define year sliding window; used in conjunction with year 2000 (see page 88).

Enhanced Profile Parameters

DU

Apart from the existing values, you can specify the following new values:

- **DU=FORCE** will force an immediate dump in the case of an abnormal termination during a NATURAL session and will terminate the NATURAL session immediately. This may be useful for testing purposes in some environments.

- **DU=SNAP** will force an immediate dump in the case of an abnormal termination during a NATURAL session; however, the NATURAL session will continue after the dump has been taken.

DYNPARM

In addition to enabling/disabling the use of dynamic profile parameters altogether, you can now also specify individual profile parameters whose dynamical specification is to be allowed/ disallowed.

FDIC, FNAT, FSEC, FSPOOL, FUSER, LFILE

In addition to the existing values, you can specify the new value "RO" to disable modifications on the file (read-only access). Thus it is possible to set read-only access for individual system files.

PC

The option CHECK (checking of "checksum" when uploading data) can no longer be specified.

The following new options can be specified:

- **OLD** — The new features of NATURAL Version 2.3 are not supported. For date and time variables, a runtime error will occur. Screens greater than 24x80 will not use the full size of the terminal buffer; 24x80 will still be used. Use OLD to avoid conflicts when working with older versions of ENTIRE Connection.

- **NEW** — The new features of NATURAL Version 2.3 are supported (this is the default).

- **NAM** — Allows you to explicitly specify that field names are to be sent when uploading/ downloading data (opposite of NONAM).

PROFILE

You can now read a profile from a system file other than the current FNAT file. To do so, you can specify the desired database ID, file number, password and cipher code with the PROFILE parameter.

RCA

With Version 2.2, it is not possible to replace statically linked subprograms.

With Version 2.3, this is possible, and an error message will be issued if a specified module cannot be loaded dynamically.

RELO

Apart from the existing values, you can specify the new value "FORCE". RELO=FORCE will force a relocation of NATURAL buffers to another storage area. This may be useful for testing purposes in some environments.

With Version 2.2, the RELO parameter can only be specified dynamically. With Version 2.3, it can also be specified statically in a parameter module.

SKEY

With Version 2.2, this parameter can only be specified dynamically. With Version 2.3, it can also be specified statically in a parameter module.

TD

With Version 2.2, you can specify the time differential in intervals of 30 minutes. With Version 2.3, you can specify it in intervals of 1 minute.

TF

For the *production-DBID* and/or *production-FNR*, you can specify an asterisk (*):

•   If you specify it for both, all production DBIDs and FNRs will be translated to the specified *test-DBID* and *test-FNR*.

•   If you specify it for the *production-FNR* only, all FNRs in the specified *production-DBID* will be translated to the specified *test-DBID* and *test-FNR*.

WORK

The functionality of this parameter has changed entirely. With Version 2.3, its functionality is the same as that of the new macro NTWORK, which in turn not only provides the functionality of the "old" WORK parameter, but also numerous new options for the definition of work files; see page 64.

XREF

Apart from the existing values, you can specify the new value "DOC". XREF=DOC corresponds to XREF=FORCE, except that no cross-reference data will be generated.

Removed Profile Parameters

The profile parameters listed below have been removed from the NATURAL parameter module.

You should not use a parameter module which contains any of these parameters. However, should these parameters still appear in a parameter module, a warning will appear, but they will be ignored and will not lead to an error.

Should you still specify any of these parameters dynamically, the following will happen (as indicated in the column "Dyn." below):

- **C** = the parameter will be converted into the corresponding new parameter;

- **E** = an initialization error will be issued;

- **I** = the parameter will be ignored.

Parameter	Comment	Dyn.
ADASVC	Replaced by new SUBSID profile parameter (see page 75).	I
AVERIO	Obsolete; the sizes of the buffers concerned are now determined automatically (see page 108).	I
BPID	Functionality now controlled by new NTBPI macro (see page 62).	C
BPRINTD	Functionality now controlled by new NTPRINT macro (see page 64).	C
BWORKD	Functionality now controlled by new NTWORK macro (see page 64).	C
BWORKDL	Functionality now controlled by new NTWORK macro (see page 64).	C
BWORKDR	Functionality now controlled by new NTWORK macro (see page 64).	C
CMPR	Obsolete; functionality no longer required.	E
CMPRTSZ	Obsolete; the size of the buffer concerned is now determined automatically.	–
EXTBUF	Obsolete; functionality no longer required.	I
FSIZE	Obsolete; the size of the buffer concerned is now determined automatically.	I

Parameter	Comment	Dyn.
GRAPHIC	Obsolete; NATURAL GRAPHICS no longer supported.	I
KC	Functionality now controlled by new NTCMPO macro (see page 63).	C
LFILMAX	Obsolete; the size of the buffer concerned is now determined automatically.	–
MCG	Functionality now controlled by new NTOPT macro (see page 63).	C
PRINTER	Functionality now controlled by new NTPRINT macro (see page 64).	C
PRTBLK	Functionality now controlled by new NTPRINT macro (see page 64).	C
SORTEOJ	Functionality now controlled by new NTSORT macro (see page 67).	C
SORTMAX	Obsolete; functionality no longer required.	I
SORTMIN	Obsolete; functionality no longer required.	I
SORTN	Functionality now controlled by new NTSORT macro (see page 67).	C
SORTOPT	Functionality now controlled by new NTSORT macro (see page 67).	C
SORTSZE	Functionality now controlled by new NTSORT macro (see page 67).	C
USIZE	Obsolete; functionality no longer required.	I
WADSIZE	Replaced by new ETPSIZE profile parameter.	E
WFOPFA	Functionality is now controlled by new NTWORK macro (see page 64).	C
WORKBLK	Functionality is now controlled by new NTWORK macro (see page 64).	C
XSORT	Obsolete; functionality no longer required.	E

New Macros

NTBPI — Buffer Pool Initialization

This macro replaces the NTBP macro for the definition of buffer pools.

NTBPI differs from NTBP in the following aspects:

- In addition to the special-purpose buffer pools, you can define the NATURAL global buffer pool. (Thus the BPID profile parameter becomes obsolete.)

- Instead of positional parameters, keyword parameters are used for the definitions of the buffer pools. This enhances the readability of the macro.

- Instead of the ADABAS SVC number, you specify the new profile parameter SUBSID (see page 75).

- Instead of the global buffer pool ID, you specify a name of up to 8 characters.

- For every buffer pool, you can specify "backup" buffer pools (see page 76).

For further information on buffer pool enhancements, see page 75.

NTCCTAB — Printer Escape Character Control Table

This macro is used to set up a table of printer control sequences, which is to be used for printing additional reports and hardcopies.

The NTCCTAB macro replaces the NTCC macro of the NATCONFG configuration module. It has the same parameter format as the Version 2.2 NTCC macro.

A new feature of the NTCCTAB macro is that you can specify a data string (up to 250 bytes) which will be sent to the printer with each *close* operation.

NTDYNP — Control Use of Dynamic Parameters

With this macro, you can control the dynamic specification of profile parameters. You can:

- disallow the dynamic specification of profile parameters altogether;
- generally allow dynamic specification, but disallow it for individual parameters;
- generally disallow dynamic specification, but allow it for individual parameters.

NTCMPO — Compilation Options

With this macro, you can set various options which affect the way in which a NATURAL object is compiled. The options correspond to those of the new system command COMPOPT (see page 34).

NTOPT — Control of NATURAL OPTIMIZER COMPILER

This macro is used to activate/deactivate the NATURAL OPTIMIZER COMPILER and controls various options related to it. Instead of the NTOPT macro, you can also use the new dynamic profile parameter OPT.

NTPRINT and NTWORK — Printer and Work File Definitions

The new macros NTPRINT and NTWORK replace the following profile parameters: BPRINTD, BWORKD, BWORKDL, BWORKDR, PRINTER, PRTBLK, WFOPFA, WORK (see page 59), WORKBLK.

Thus, all print and work file definition items are now contained in two macros, instead of being spread over numerous parameters. This will make the definition of print/work files easier and more comfortable.

In addition to taking over the functions that with Version 2.2 are provided by the above parameters, the macros NTPRINT and NTWORK (as well as the corresponding new dynamic profile parameters PRINT and WORK) provide the following enhancements for the definition of print/work files:

- You can define logical file names (with Version 2.2, the file names are CMPRT*nn* and CMWKF*nn* respectively; these will continue to be the default names).

- You can determine for each print/work file individually when it is to be opened: at the start of the NATURAL session, at the start of the program referencing the file, or when the file is first accessed by a corresponding statement.

- You can determine for each print/work file individually when it is to be closed: at the end of the program referencing the file, when command mode is reached, or at the end of the NATURAL session.

- For batch and TSO environments, you can pre-define for each print/work file individually the default record format, block size and record length to be used. These values will be used if no corresponding values are supplied by the operating system. It is no longer necessary to supply JCL for print/work files; they can be allocated dynamically by any NATURAL program before they are first accessed (see also page 77).
The print/work files defined in the JCL are only used automatically by AM=STD if no definitions for another access method for these files exist in the parameter module (or dynamically).

- You can determine the action to be taken if a record exceeds the defined record length: whether an error is to be issued or the record is to be truncated.

- You can pre-define values for the print file options PROFILE, FORMS, NAME, DISP and COPIES that can be specified with a DEFINE PRINTER statement.

Note: With Version 2.2, the NATURAL ADVANCED FACILITIES user profile name is specified with the PRINTER profile parameter. As the PRINTER parameter becomes obsolete with Version 2.3, the new profile parameter NAFUPF is provided for the specification of the NATURAL ADVANCED FACILITIES user profile name.

NTRPC — Remote Procedure Call Options

In this macro, you can specify several options which affect the handling of NATURAL remote procedure call (RPC).

With Version 2.2, most of these options are specified either with NATSRVI when the server is started or in the Parameter Section of the SYSRPC utility; another option (SIZE) replaces the EXTBUF and DSIZE profile parameter specifications.

The individual options that are available are described in the *NATURAL Installation and Operations Manual for Mainframes*.

Moreover, the new profile parameter RPC allows you to specify dynamically the same options that can be set in the NTRPC macro.

NTSCTAB — Overwriting of Scanner Character Type Table

This macro allows you to overwrite the definitions in the scanner character type table NTSCTAB as contained in the configuration module NATCONFG. The NTSCTAB table defines the properties of characters used in mask definitions for the SCAN/MASK functions.

NTTAB — Overwriting of Standard Output Translation Table

This macro allows you to overwrite the definitions in the translation table NTTAB as contained in the configuration module NATCONFG. The NTTAB table is the standard output translation table.

Instead of the NTTAB macro, you can also use the new dynamic profile parameter TAB.

NTTABA1 and NTTABA2 —
Overwriting of EBCDIC-ASCII Translation Tables

These macros allow you to overwrite the definitions in the translation tables NTTABA1 and NTTABA2 as contained in the configuration module NATCONFG. The NTTABA1 table is used for EBCDIC-to-ASCII translation; the NTTABA2 table is used for ASCII-to-EBCDIC translation.

Instead of the macros NTTABA1 and NTTABA2, you can also use the new dynamic profile parameters TABA1 and TABA2 respectively.

NTTABL — Overwriting of SYS Library Output Translation Table

This macro allows you to overwrite the definitions in the translation table NTTABL as contained in the configuration module NATCONFG. The NTTABL table is used to translate output produced by programs located in "SYS" libraries.

Instead of the NTTABL macro, you can also use the new dynamic profile parameter TABL.

NTTAB1 and NTTAB2 —
Overwriting of Alternate Output/Input Translation Tables

These macro allow you to overwrite the definitions in the translation tables NTTAB1 and NTTAB2 as contained in the configuration module NATCONFG.

The NTTAB1 table is the alternate output translation table for the secondary character set used when the profile/session parameter PM=C is set. The NTTAB2 table is the alternate input translation table for the secondary character set used when the profile/session parameter PM=C is set.

Instead of the macros NTTAB1 and NTTAB2, you can also use the new dynamic profile parameters TAB1 and TAB2 respectively.

NTSORT — Control of Sort Program

With Version 2.2, the various functions that control the sort program used by NATURAL for SORT statements are determined by various profile parameters and installation options.

With Version 2.3, the new macro NTSORT combines all these functions, thus allowing you to set in a single location all options which control the handling of the sort program used by NATURAL when a SORT statement is executed.

The following options are available:

- **WRKSIZE** — Determines the size of the work buffer to be used by the sort program (corresponds to the Version 2.2 profile parameter SORTSZE).

- **STORAGE** — Determines the type of storage medium to be used by the incore sort: main storage, a sort buffer pool, or COM-PLETE SD files. (With Version 2.2, this is determined by an installation option specified at the linking of the NATURAL nucleus.)

- **EXT** — Determines if an external sort program is to be used or not. (With Version 2.2, this is determined by an installation option specified at the linking of the NATURAL nucleus.)

- **EXTNAME** — Specifies the name of the external sort program to be used (corresponds to the Version 2.2 profile parameter SORTN).

- **EXTOPT** — Allows you to specify additional options for the external sort program (corresponds to the Version 2.2 profile parameter SORTOPT).

- **EXTEOJ** — Determines the action to be taken if an error is detected during the execution of the external sort program (corresponds to the Version 2.2 profile parameter SORTEOJ).

Moreover, the new profile parameter SORT allows you to specify dynamically the same options that can be set in the NTSORT macro.

NTUSER — Use of Alternative Parameter Module

This macro allows you to restrict the use of a parameter module as an alternative parameter module (as specified with the dynamic profile parameter PARM).

In a parameter module, you specify with the NTUSER macro the IDs of the users who are to be allowed to use the parameter module as an alternative parameter module. Only the specified users will then be allowed to specify the name of that parameter module with the PARM parameter.

NTUTAB1 and NTUTAB2 — Overwriting of Case Translation Table

These macros allow you to overwrite the definitions in the translation tables NTUTAB1 and NTUTAB2 as contained in the configuration module NATCONFG. The NTUTAB1 table is used for lower-to-upper-case translation; the NTUTAB2 table is used for upper-to-lower-case translation.

Instead of the macros NTUTAB1 and NTUTAB2, you can also use the new dynamic profile parameters UTAB1 and UTAB2 respectively.

Enhanced Macros

NTDB

In addition to the database type, you can now specify the following options for the databases:

- READ/NOREAD: determines whether read-only access to a database is permitted or not,

- OPEN/NOOPEN: determines whether a database open is to be performed or not.

Moreover, you can now specify whether a database is to be accessed or not by various SOFTWARE AG products.

NTFILE

Apart from the existing values, you can specify the new value "RO" to disable modifications on the file (read-only access). Thus is it possible to set read-only access for individual system files.

Removed Macro

NTBP

The NTBP macro can no longer be used in a NATURAL parameter module. It has been replaced by the new NTBPI macro (see page 62).

DATABASE INTERFACES

This chapter contains information related to:

- ADABAS,
- DB2,
- DL/I,
- VSAM.

ADABAS

The following ADABAS features are supported with NATURAL Version 2.3:

- Support of null values in analogy to SQL databases: For database fields that can contain null values, corresponding null-value indicator fields are generated in the DDM. These null-value indicator fields are evaluated when a FIND or READ statement is executed, and can be queried in a FIND statement.

- Support of the "read backwards" feature (see also page 25).

- Support of repositioning within database loops (see also page 25).

- The maximum number of occurrences of a periodic group has been increased from 99 to 191.

- Support of 2-byte database IDs and file numbers (as introduced with ADABAS Version 6).

- Access to the database has been improved by search buffer enhancements for a database access containing the operators NOTEQUAL and OR. This will contribute to better performance in this area.

- Support of the "security by value" feature (this is activated/deactivated via the new NATURAL profile parameter ADASBV).

- Enhanced error texts, including: database ID, file number, subcode of ADABAS response code (if available), and NET-WORK node ID (if available).

- If ADABAS Version 6.2 (or above) link routines and SVC are used for MVS batch mode, CICS or IMS/TM, ADABAS will assume NATURAL to be two users: one for system calls of the NATURAL runtime system, and one for the actual application calls. Thus it is possible, in almost all cases where a system call receives an ADABAS time-out code (NAT3009), to restart the ADABAS transaction without having to inform the user of this.

DB2

The following DB2-related enhancements are provided with NATURAL Version 2.3:

- Support of the JOIN syntax in the FROM clause of the SELECT statement.

- Support of "subselects" in the FROM clause of the SELECT statement.

- Support of the AS clause, that is, the specification of a name for result columns, in the *selection* list.

- Support of the "WITH *isolation-level*" clause, that is, the specification of the isolation level for individual statements. In addition, a new isolation level "UR" (uncommitted read) can be specified for the SELECT statement.

- Support of DB2 "stored procedures", which can be invoked via the PROCESS SQL statement.

- The function "Retrieval of System Tables" of the SYSDB2 tools has been enhanced to display new columns supported by DB2 Version 4.

- The function "Catalog Maintenance" of the SYSDB2 tools has been enhanced by the new clauses supported by DB2 Version 4. This includes in particular the CREATE/ALTER TABLE syntax: user-defined defaults; check constraints; shorthand notation for primary, unique and foreign key.

- The SYSDB2 tools have been enhanced to allow you to comfortably define DB2 "stored procedures".

Together with Version 2.3.1 of NATURAL, Version 2.4.1 of NATURAL for DB2 is released.

See also the *NATURAL for DB2 Version 2.4.1 Release Notes*.

DL/I

The following DL/I-related enhancements are provided with NATURAL Version 2.3:

- Fast Path support.

- Repair of invalid DL/I data in numeric/packed fields.
  Repair can be switched on/off using NDLPARM.
  Warning message can be switched on/off using NDLPARM.

- DLILOG displays the number of SSAs for each DL/I call.
  This allows you to determine on which level the segment is accessed.

- NATURAL for DL/I addresses are relocated only if necessary.
  This relocation can be switched on/off using NDLPARM.

- Provide environment-independent NATURAL for DL/I nucleus, which can be linked in the LPA/SVA.

- Provide fast locate buffer for buffer pool calls.

- Support buffer pool blacklist feature for NATURAL for DL/I objects.

- Allow NATURAL for DL/I objects to be resident or not.
  Residency can be switched on/off using NDLPARM.

- Re-use I/O areas as often as possible so as to reduce the number of GETMAIN and FREEMAIN requests. The initial size of the I/O areas is defined using NDLPARM.

- Use DL/I buffer pool (if defined) instead of NATURAL program buffer pool.

Together with Version 2.3.1 of NATURAL, Version 2.3.1 of NATURAL for DL/I is released.

The NATURAL DL/I interface, which used to be described in the *NATURAL Installation and Operations Manual for Mainframes*, is now described in a separate manual: the *NATURAL for DL/I Manual*.

VSAM

The following VSAM-related enhancements are provided with NATURAL Version 2.3:

- Support of global format IDs. This reduces the amount of CPU time required for command analysis.

- Support of path access for VSAM, to be compatible with Cobol applications.

- Support of VSAM RLS (record-level sharing). This permits Sysplex-wide VSAM access.

- Support of the "read backwards" feature (see also page 25).

- Support of repositioning within database loops, using "skip sequential" processing (see also page 25 and page 29).

- Performance improvements under IMS/TM by using global shared resources (GSR); thus a CLOSE/OPEN of ACBs at each terminal I/O is avoided.

Together with Version 2.3.1 of NATURAL, Version 2.4.1 of NATURAL for VSAM is released.

For the time being, Version 2.4.1 of NATURAL for VSAM is only available under MVS. Under VSE, it will be made available as soon as possible after the release of NATURAL 2.3.1.

6

OPERATING SYSTEM AND TELEPROCESSING INTERFACES

This chapter contains information related to:

- full support of 31-bit mode
- dynamically loaded shared nucleus
- buffer pool
- work files and print files
- changed layout of control blocks
- storage management
- support of LE/370 (under MVS only)
- support of Sysplex (under MVS only)
- BS2000
- CICS
- COM-PLETE
- IMS/TM
- MVS/ESA
- TSO
- UTM
- VSE

Full Support of 31-Bit Mode

NATURAL 2.3 provides full support of 31-bit addressing mode. This means that NATURAL can now be loaded either below or above 16 MB in all supported environments. Moreover, most NATURAL buffers are by default allocated above 16 MB.

Dynamically Loaded Shared Nucleus

The new profile parameter NUCNAME allows you to specify the name of the (environment-independent) shared nucleus if it is to be loaded dynamically and not linked to the environment-dependent nucleus.

As this parameter can be specified dynamically, it allows you to use different shared nuclei (for example, for production and for testing) together with the same environment-dependent nucleus without having to relink the nucleus.

Buffer Pool

Elimination of ADABAS SVC (under MVS only)

For the use of the global buffer pools (NATURAL buffer pool, sort buffer pool, editor buffer pool), the ADABAS SVC is no longer required (and the ADASVC profile parameter has become obsolete).

Instead, NATURAL Version 2.3 uses the name/token callable services of MVS/ESA to anchor the directory of all global buffer pool addresses used by NATURAL. Consequently, global buffer pools are only available under MVS/ESA or above.

Propagation of Global Buffer Pool Changes (under MVS and BS2000)

With Version 2.3, if a NATURAL object residing in a (global or local) buffer pool is modified, it is possible to propagate this modification to all other global buffer pools; that is, the next time the object is requested, it will be newly loaded from the system file.

This propagation is optional and can take place within a single operating-system environment. Under MVS, the propagation can also be done within a Sysplex environment, using the XCF signalling services (see page 79).

The propagation is controlled by the new profile parameter BPPROP. It allows you to propagate changes to all global buffer pools within the same BS2000 or MVS system, to all global buffer pools within an MVS Sysplex, or not at all.

Name of NATURAL Global Buffer Pool

With Version 2.3, the name of the NATURAL global buffer pool is specified with the new NATURAL profile parameter BPNAME (which replaces the Version 2.2 profile parameter BPID).

Search Sequence for Objects in Buffer Pool and Database

The new NATURAL profile parameter BPSFI allows you to determine the sequence in which a requested object that is to be executed is searched for in the buffer pool and in the system file(s) on the database.

You can choose between two search sequences (NATURAL Version 2.2 always uses Sequence 1):

Sequence 1 — alternating search for each library:

① The *buffer pool* is searched for the object in the current library.
  If it is not found there, the *database* is searched for the object in the current library.

② If it is not found there either, one steplib after another is searched in the same sequence; that is, for each steplib, first the *buffer pool* is searched, then the *database*.

Sequence 2 — search buffer pool first for all libraries, then database:

① The *buffer pool* is searched for the object
  (first in the current library, then in one steplib after another).

② If the object is not found anywhere in the buffer pool, the *database* is searched for it
  (first in the current library, then in one steplib after another).

Dynamic Buffer Pool Initialization

With Version 2.2, it is not possible to make buffer pool specifications dynamically.

With Version 2.3, the new profile parameter BPI allows you to specify NATURAL buffer pools dynamically. It offers the same functionality as the new macro NTBPI (see page 62). For the primary NATURAL buffer pool, the dynamic parameters BPNAME, BPLIST, BPSIZE and BPTEXT can also be used.

Backup Buffer Pools

For every buffer pool, you can specify one or more backup buffer pools; that is, alternative (global or local) buffer pools which will be used in case the buffer pool is not available at session initialization or cannot be allocated. Also, if the currently used buffer pool is stopped during the NATURAL session, NATURAL will continue to work with one of the backup buffer pools without the session being terminated.

The specification of the backup buffer pools is made in the new NTBPI macro or with the corresponding new BPI profile parameter (see also page 62).

Work Files and Print Files

Dynamic Allocation/Release of Work/Print Files

With Version 2.2, all work files and print files that might possibly be used have to be allocated in the JCL/JCS used to start NATURAL, regardless of whether they were actually used or not.

With Version 2.3, a CALL interface is provided, for batch mode (MVS, BS2000) and single-user mode (TSO, TIAM, CMS) respectively, which allows you to dynamically allocate and release work and print files as needed. This also allows you to print the contents of a print file without having to end the NATURAL session.

Centralized and Enhanced Definition of Work/Print Files

With Version 2.2, the various items for the definition of print and work files are determined by various profile parameters.

With Version 2.3, instead of being spread over numerous parameters, all print and work file definition items are contained in two new parameter module macros, NTPRINT and NTWORK. This makes the definition of print/work files easier and more comfortable.

In addition, these two macros provide several enhancements for the definition of print/work files; see page 64.

Hardcopy Access Method

The new NATURAL profile parameter HCAM allows you to determine which access method is to be used for hardcopy output processing: standard batch, COM-PLETE, CMS or NATURAL ADVANCED FACILITIES.

Changed Layout of Control Blocks

With Version 2.3, the layout of the NATURAL control blocks (BB, IOCB, etc.) has been changed. Consequently, any user exits or user routines that use these control blocks have to be adjusted.

Storage Management

Work Pool Size

The new NATURAL profile parameter WPSIZE controls the sizes of physical GETMAIN storage pools below and above the 16 MB line. With the WPSIZE parameter, you specify two values:

- The first value defines the work pool size below the 16 MB line (in KB). The upper limit is 1024. The default is 32.

- The second value defines the work pool size above the 16 MB line (in KB). The upper limit is 16384. The default is 128.

NATURAL will allocate the work pools according to the specified values.

If "0" is specified, no dynamic work pool will be allocated and all physical GETMAIN requests will be passed to the operating system.

Buffers for Data Areas

With Version 2.2, the sizes of buffers for global and local data areas was determined by the NATURAL profile parameters ESIZE and DATSIZE respectively.

With Version 2.3, the values of these parameters are only used as initial values for the corresponding buffers. During the NATURAL session, the sizes of these buffers are increased automatically as required.

Support of LE/370 (under MVS only)

As of Version 2.3, NATURAL under MVS supports IBM LE/370 Language Environment subprograms. NATURAL prepares the CALL statement which invokes these subprograms to make them executable under LE/370. LE/370 subprograms can be static or dynamic subprograms of NATURAL. For details, see the *NATURAL Installation and Operations Manual for Mainframes*.

Support of Sysplex (under MVS only)

Sysplex in this context means the parallel Sysplex that was introduced by MVS/ESA SP5.1.

With Version 2.3, NATURAL makes use of the full Sysplex functionality as provided by the TP monitors CICS (as of CICS/ESA Version 4.1) and IMS/TM (as of Version 6.1).

Thus, it is possible for the different transaction steps of a NATURAL session to be executed in different address spaces of the the same MVS image, or even of different MVS images.

Another benefit is that it is possible to continue a NATURAL session after an IPL and a warm/emergency restart of IMS/TM.

Moreover, Sysplex support makes it possible to support the IMS/TM extended restart facility (XRF).

NATURAL's Sysplex support is provided primarily by enhancements to:

- the NATURAL nucleus,

- rolling of the NATURAL threads,

- the NATURAL buffer pool.

Note: If your NATURAL system files are stored in an ADABAS database, Sysplex support requires ADABAS Version 6.2.1 (or above) link routines and SVC.

Nucleus

As the "life cycle" of a NATURAL session may be spread over different address spaces and/or MVS images, it is possible for the NATURAL nucleus to reside at a different virtual address within each address space.

Rolling

It is necessary to pass the NATURAL thread (that, is the NATURAL session-related data) from one transaction step to the next.

A roll file allocated to a shared DASD would enable the Sysplex support in this area, but this would not be a viable solution with regard to performance aspects.

Therefore, a *roll server* is provided with NATURAL 2.3 under CICS and IMS/TM.

For synchronization purposes, the roll server uses the Coupling Facility to which the roll-file directory is allocated.

Internally, the roll server uses a roll buffer allocated within a data space. When data are written to the terminal, the NATURAL thread is moved into the roll buffer, and written asynchronously to the roll file. At terminal input, the NATURAL thread is retrieved either directly from the roll buffer, or — if the preceding write operation was executed in a different MVS image — rolled in from the roll file.

If all application-owning regions (AORs) or all CICS images are located in the same MVS image, the Coupling Facility will not be used (except if a NATURAL thread has to be written to the roll file because there is not enough space left in the roll buffer).

An authorization is needed in order to access to the Coupling Facility, and the number of simultaneous connections to the Coupling Facility is limited to 32; therefore the roll server is separated from the NATURAL nucleus and located in its own address space. The roll server has to be started in each MVS image within a Sysplex. The roll server could thus be termed a "multi-systems application".

It is also possible to use the roll server within a single MVS image.

Buffer Pool

The NATURAL global buffer pool is accessible from within all address spaces in a specific MVS image. As it is possible for a NATURAL session to span different MVS images, a Sysplex-wide global buffer pool allocated to the Coupling Facility is needed.

For good performance, a global buffer pool should be as large as possible; however, the available space in the Coupling Facility is rather limited. Morever, the maximum size of an item allocated to the Coupling Facility is limited to 64 KB — a size which a NATURAL object may easily exceed. Therefore it is necessary to allocate multiple global buffer pools, one in each MVS image (using the same buffer pool name).

With NATURAL 2.3, it is possible to propagate object modifications from one (local or global) buffer pool to all global buffer pools that are allocated either within the same MVS image or in all MVS images contained in the Sysplex.

For example, if a NATURAL program is newly cataloged in one MVS image, the program will be deleted from all other global buffer pools in the Sysplex, thus avoiding that the old program can still be executed.

To activate the propagation of object modifications from one buffer pool to other buffer pools, the new profile parameter BPPROP is provided (see also page 75).

BS2000

NATURAL Version 2.3 requires BS2000 Version 10 or above.

The following enhancements are provided with NATURAL Version 2.3:

- support of RCA (resolve CSTATIC addresses) technique in all environments (batch mode, TIAM and UTM);

- display of all allocated NATURAL common memory pools;

- environment-independent NATURAL nucleus which can be shared between batch mode, TIAM and UTM;

- the generation of the batch-mode driver (which used to be done by assembling NATBS2) is now done via a macro; thus, you no longer have to modify the source of the batch driver;

- the NATURAL/BS2000 batch nucleus has been split into a front-end part and an environment-independent reentrant part;

- the assembled macro BS2STUB must be linked to the front-end part of the batch, TIAM and UTM drivers; functions of BS2STUB: enabling of all common memory pools, dynamic loading of 3GL programs, execution of BS2000-specific macro calls.

The following restrictions apply with NATURAL Version 2.3:

- Third-language programs invoked by NATURAL which either run under CRTE or use ILCS must have the same AMODE as NATURAL.

- The device type "8160" is no longer supported.

CICS

NATURAL Version 2.3 requires CICS/MVS Version 2.1 (or above) or CICS/VSE Version 2.1 (or above) respectively.

The following enhancements are provided with NATURAL Version 2.3:

- The use of NATURAL work files is also available under CICS.

- CICSPlex in a parallel Sysplex is supported (requires CICS Version 4.1 or above).

- With Version 2.2, the maximum number of CMROLL calls after which a roll-out of the NATURAL thread is to be performed is determined by the MAXROLL parameter in the NCIPARM macro. With Version 2.3, it is determined by the new NATURAL profile parameter MAXROLL.

- Permanent threads (formerly program storages) are now allocated via SVC GETMAIN or CICS GETMAIN SHARED.

COM-PLETE

NATURAL Version 2.3 requires COM-PLETE Version 4.6 or above.

The following enhancements are provided with NATURAL Version 2.3:

- Instead of being loaded into the COM-PLETE thread, the NATURAL/COM-PLETE bootstrap module is loaded as a resident page program. Thus there will be more space available in the COM-PLETE thread for NATURAL storage requirements.

- For better performance, the editor work file is no longer a VSAM file, but an SD file. Consequently, it is no longer possible for COM-PLETE to share a global editor buffer pool with other environments (for example, CICS).

- With Version 2.2, the maximum number of CMROLL calls after which a roll-out of the NATURAL thread is to be performed is determined by the MAXROL parameter in the NTCOMP macro. With Version 2.3, it is determined by the new NATURAL profile parameter MAXROLL.

- If you issue the terminal command %H without a destination, the default printer specified with the NATURAL profile parameter HCDEST is used. With Version 2.2, %H displays a window in which you can overwrite this default destination; with Version 2.3, this window is no longer displayed.

IMS/TM

The NATURAL IMS/TM interface is not yet available with Version 2.3.1. It will be available with Version 2.3.2.

MVS/ESA

The NATURAL Subsystem

The NATURAL subsystem is only available under MVS/ESA. A NATURAL subsystem consists of the following components: one or more global buffer pools, an authorized services manager, and a roll server.

The NATURAL subsystem is identified by the new NATURAL profile parameter SUBSID and by corresponding startup parameters for above components.

Via the NATURAL subsystem technique, multiple roll servers can be used simultaneously and multiple independent sets of global buffer pools can be created — in fact, multiple NATURAL runtime environments can be created which will be totally independent from one another.

TSO

The following enhancement is provided with NATURAL Version 2.3:

- You can issue a TSO command from within NATURAL.

UTM

The following enhancements are provided with NATURAL Version 2.3:

- Data exceptions are intercepted and error NAT0954 will be issued.

- The new swap pool manager supports up to 15 logical swap pools within a single physical swap pool. This will reduce/eliminate reorganization requirements.

- The swap pool directory can be made resident page; this will reduce/eliminate paging.

- The new BS2000 feature FASTPAM (with forward-eventing) is provided as an alternative access method for the NATURAL roll file.

- The NATURAL communication area (Kommunikationsbereich, KB) will be saved before executing a PEND PR(ogram), and restored after return from UTM.

- The terminal control table is located in the swap pool. This will reduce the I/O rate and storage requirements.

- It is possible to write data asynchronously to the swap file even if no swap pool is used (with Version 2.2, they can only be written synchronously if no swap pool is used).

VSE

The following enhancements are provided with NATURAL Version 2.3:

- The datasets CMSYNIN and CMOBJIN are file-independent. This makes it possible to read NATURAL input from disk file.

- The "STXIT AB" are active for all file I/Os to intercept I/O errors and to issue NATURAL error messages.

- It is possible to direct an output work file to PUNCH.

MISCELLANEOUS

This chapter contains information on:

- user interface,

- support of year 2000,

- NATURAL remote procedure call (RPC),

- search sequence for objects to be executed,

- size of object code,

- size of data areas,

- number of variables in programming objects,

- evaluation of printer definitions,

- arithmetic functions,

- array operations with variable index ranges,

- session control,

- error messages,

- database access,

- terminal I/O handling,

- handling of protected fields,

- suppress zero display for time fields,

- loading of datasets with INPL,

- support of Kanji printing,

- improved handling of internal buffers,

- SOFTWARE AG editor,

- dump analysis tool SYSNDA,

- NATURAL OPTIMIZER COMPILER,

- ENTIRE Connection.

User Interface

The NATURAL Main Menu and its subordinate menu screens have been revised so as to provide you with a better structured user interface.

The new menu structure is described in Chapter 1 of the *NATURAL User's Guide for Mainframes*.

Note: In batch mode, the Main Menu can no longer be used. In batch mode, a function from the Main Menu's subordinate menus can be invoked via the corresponding system command.

Support of Year 2000

The "Year 2000" Problem

Numerous applications use a 2-digit format instead of a 4-digit format for the representation of year information. This means that, for example, 24th December 1996 is represented as "24-12-96". This might lead to misinterpretations of dates in the next century, because 24th December 2000 would be represented as "24–12–00" — which could not be distinguished from the representation of 24th December 1900.

The following sections describe what is provided with NATURAL Version 2.3 to help solve this problem.

In addition, Chapter 5 of the *NATURAL Programming Guide* contains a new section on the processing of date information.

Handling Date Information with IS Option and VAL Function

With Version 2.2, the IS option, which is used to check whether the contents of an alphanumeric field can be converted to another format, can only be applied to format D if the value checked contains 2-digit year information. The mathematical function VAL, which is used to extract a numeric value from an alphanumeric field, only accepts date information that corresponds to the rules for the IS option; this means that a date value which contains 4-digit year information cannot be extracted.

With Version 2.3, the IS option and the VAL function accept both 2- and 4-digit year information.

Default Edit Mask for Date — The DF Parameter

If the value of a date field is converted to alphanumeric format (for example, in a MOVE, DISPLAY, PRINT, WRITE or INPUT statement) and no edit mask is specified for the conversion, the default date format as determined by the profile parameter DTFORM is used as edit mask, but only providing 2 digits for the year information. This means that even if the date value contained the century, this information would be lost during the conversion.

The same is true for the input validation of a date variable used in an INPUT statement. If no edit mask is specifed, the input is validated according to the default date format determined by the DTFORM parameter, with 2 digits for the year information.

With Version 2.2, both the above effects can be avoided by explicitly specifying edit masks. However, a change of the format of alphanumeric date representations would then no longer be possible by simply changing the setting of the DTFORM parameter; instead, it would be necessary to adjust the specified edit masks in the programs.

With Version 2.3, the new session parameter DF allows you to specify whether the length of a date when converted to alphanumeric representation is to be 8 or 10 characters. The DF parameter can be specified with the FORMAT statement and various other statements (at statement and field level), and it is evaluated at compilation. This allows you to gradually change your applications to use 4-digit year representations and at the same time continue to make use of the flexibility provided by the DTFORM parameter.

Date System Variables

The NATURAL date system variables *DATD, *DATE, *DATI, *DATJ and *DATU, which contain the current date in various formats, all provide the year information as 2-digits.

With Version 2.3, corresponding new date system variables providing 4-digit year information are available (see page 12).

Date as Selection Criterion in Utilities

The NATURAL utilities NATUNLD, NATLOAD and SYSMAIN allow you to specify a date as selection criterion.

With Version 2.2, the date can only be specified with 2 digits for the year, and the current century is internally appended to the specified date.

With Version 2.3, the above utilities accept dates with both a 2- and a 4–digit year specification as selection criterion.

"Sliding Window" — The YSLW Parameter

With the new profile parameter YSLW, you can set a so-called "sliding window", which allows you to continue using 2-digit year representations and at the same time ensure that any 2-digit year value can be uniquely related to a specific century.

Date Format for Output — The DFOUT Parameter

The new session/profile parameter DFOUT allows you to control the format in which date fields are output with INPUT, DISPLAY, PRINT and WRITE statements.

For date fields which are displayed with the above statements and for which neither an edit mask is specified nor a DF parameter applies, the DFOUT parameter determines the format in which the field values are displayed:

- with a 2-digit year component and delimiters,
- with a full 4-digit-year component and no delimiters.

The lengths of the date fields are not affected by the DFOUT setting, as either date value representation fits into an 8-byte field.

The DFOUT parameter can be set in the NATURAL parameter module, dynamically when NATURAL is invoked, or with the system command GLOBALS. It is evaluated at runtime.

Date Format for Stack — The DFSTACK Parameter

The new session/profile parameter DFSTACK allows you to control the format in which the values of date variables are placed on the stack (via a STACK, RUN or FETCH statement). Three options are possible:

- Date values are stacked with a full 4 digit-year component and no delimiters.
- Date values are stacked with a 2-digit year component and delimiters.
- Date values are stacked with a 2-digit year component and delimiters; in addition, a change in the century will be intercepted: when the value is read from the stack, the century is assumed to be the current one or determined by the the YSLW parameter (see above); if this leads to the century being different from that of the original date value, a runtime error will be issued.

The DFSTACK parameter only applies to date fields for which no DF parameter is specified. It can be set in the NATURAL parameter module, dynamically when NATURAL is invoked, or with the system command GLOBALS. It is evaluated at runtime.

NATURAL Remote Procedure Call (RPC)

Prerequisite

The RPC functionality of NATURAL Version 2.3.1 requires ENTIRE BROKER ACI Version 2.1.2 (or above).

New Macro NTRPC in Parameter Module

The NATURAL parameter module provides a new macro, NTRPC, in which you specify several options which affect the handling of NATURAL RPC (see page 65).

Programming Language Enhancements for Conversational RPCs

With Version 2.3, conversational RPC support is provided for client/server communication (support of remote CALLNAT).

For conversational RPCs, the following enhancements to the NATURAL programming language are available:

- **OPEN CONVERSATION** — This new statement allows the client to get a server for exclusive use to execute a number of services (subprograms) within one server process. This exclusive use is called conversation. With OPEN CONVERSATION, you open a conversation and specify the names of the subprograms which are to be involved in this conversation. OPEN CONVERSATION will assign a unique ID that identifies the conversation to the system variable *CONVID (see below).
  Several conversations can be open at the same time. To switch from one open conversation to another, you assign the corresponding conversation ID to *CONVID.

- **CLOSE CONVERSATION** — This new statement allows the client to close conversations. You can close the current conversation, a specific other open conversation, or all open conversations (see also page 11).

- **DEFINE DATA CONTEXT** — The new CONTEXT clause of the DEFINE DATA statement is used to define the data that are to be available to several subprograms within a conversation; see page 90 for details.

- ***CONVID** — This new system variable contains the conversation ID of a conversational RPC. The value of *CONVID is set by the OPEN CONVERSATION statement. NATURAL RPC will examine the current conversation for the subprogram to be executed remotely and will pass it to the appropriate server process.

DEFINE DATA CONTEXT

With Version 2.3, it is possible for several remote subprograms within one conversation to share the same data without having to explicitly pass these data as parameters. The fields to be available to the subprograms are defined as *context variables* in a DEFINE DATA CONTEXT statement in each subprogram in which they are to be available.

A context variable is referenced by its name, and its content is shared by all subprograms referring to that name within one conversation.

Each conversation has its own set of context variables. Context variables cannot be shared between different conversations.

The context variables will be reset to their initial values when an OPEN CONVERSATION statement is executed or a single (non-conversational) remote CALLNAT is performed.

Note: One of the great advantages of using NATURAL for client/server applications is that you can develop and test your applications locally, and then distribute them for production. This requires that subprograms using context variables behave the same way whether they are executed locally or remote. For remote conversations, this is given, as one server may have only one open conversation at a time and consequently all context variables belong to this conversation; for local conversations, NATURAL precludes any confusion between context variables of the same names belonging to different conversations (by internally identifying the variables not only by their names but also by their respective conversation IDs).

RPC Without Stub

With Version 2.2, you have to generate a stub for every subprogram to be executed remotely.

With Version 2.3, if a stub for a remote subprogram does not exist, NATURAL RPC will automatically generate the necessary data which would normally be supplied by the stub — and invoke the remote subprogram as if a stub existed for it. This means that you no longer need to generate stubs for remote subprograms.

Nonetheless it is possible to continue to use existing stubs.

Remote Directory

With Version 2.2, the locations (server addresses) of the remote subprograms have to be defined in a local directory for each RPC client application. For multiple clients using the same remote subprograms, this means that identical directory information appears — and has to be maintained — on every single client.

With Version 2.3, you can define one central remote directory on the server; this remote directory can be accessed by all clients. This drastically reduces the maintenance effort of directory information. Moreover, the remote directory provides a single central place of reference to all services available in your client/server environment.

The remote directory server is implemented as a NATURAL subprogram. A sample of this subprogram, named RDSSCDIR, is provided in the library SYSRPC. It reads the required directory information from a work file. The interface of the subprogram is documented so that you can develop your own remote directory service.

If a remote subprogram is not found in the local directory, it will be sought in the remote server directory (by executing an internal remote CALLNAT). An internal directory cache minimizes the access to the remote directory. The cache information is controlled by an expiration date set by the remote directory server.

Remote Error Handling

With Version 2.2, NATURAL errors that occur in remote subprograms on the server are handled directly by the error transaction at level 1 of the application, to which the actual error information is not always available.

With Version 2.3, errors that occur in remote subprograms on the server are handled in the same way as errors in subprograms invoked locally; that is, the same ON ERROR handling mechanisms apply.

Reduced Data Transfer Load

With Version 2.3, the AD= specifications of CALLNAT parameters (see also page 14) is also evaluated for remote subprograms in a client/server environment:

- AD=M parameters are passed from the client to the server and back again.
- AD=O parameters are only passed from the client to the server, but not back.
- AD=A parameters are only passed from the server back to the client.

This reduces the load of data to be sent to a minimum.

Support of NATURAL SECURITY

With Version 2.2, a remote subprogram will always be sought in the current library on the server; however, the client has no control over which library is the current one on the server when the remote CALLNAT is performed.

With Version 2.3, NATURAL RPC also supports NATURAL SECURITY in client/server environments, and at the same time allows clients to set the current library on a server.

NATURAL RPC's Server Maintenance on the client provides a new option "Logon", which can be set for individual servers, or for a node so as to apply to all servers belonging to that node.

If the "Logon" option is set to "yes" and a remote CALLNAT is performed, the user's ID and password as well as the current library ID from the client will be passed to the server along with the CALLNAT request.

The user ID and password are established as follows:

- If the client runs under NATURAL SECURITY, the user ID and password from the NATURAL SECURITY logon on the client will be encrypted into a security token and passed to the server.

- For non-NATURAL SECURITY clients, a user exit will be provided which the user has to execute and which will prompt the user to specify a user ID and password — which will then be passed to the server.

The server will verify the user ID and password; and before executing the requested subprogram, NATURAL will then perform a logon on the server using the current library ID from the client.

If the server runs under NATURAL SECURITY, the user ID and password from the client will be verified against the corresponding user security profile on the server, and the logon to the requested library and the execution of the subprogram will be performed according to the corresponding NATURAL SECURITY library and user profile definitions on the server.

After the execution of the subprogram, the library used before the CALLNAT request will be made current again on the server.

In the case of a conversational RPC (see also page 89), the first CALLNAT request within the conversation will set the library ID on the server; and the CLOSE CONVERSATION statement will reset the library ID on the server to the one before the first request of the conversation.

For compatibility, Version 2.3 servers still support remote CALLNATs from client where the "Logon" option is not set.

To enforce the "Logon" option — that is, if you want a server to accept only requests from clients where the "Logon" option *is* set — an option in the new parameter module macro NTRPC (see page 65) is provided, which will be evaluated when you start the server.

Search Sequence for Objects to be Executed

Search for Objects to be Executed from User Libraries

The sequence in which user-written NATURAL objects that are to be executed from the FUSER system file are searched for, has been enhanced: if an object cannot be found on the FUSER file, the library SYSTEM on the system file FNAT is also searched for it. The search sequence is:

1. current library (as defined by system variable *LIBRARY–ID),

2. steplibs (in sequence as specified in NATURAL SECURITY profile of current library),

3. default steplib (as defined by system variable *STEPLIB),

4. library SYSTEM on FUSER system file,

5. library SYSTEM on FNAT system file (← *new!*).

Search for Objects to be Executed from System Libraries

The sequence in which NATURAL objects that are to be executed from the FNAT system file are searched for, has been enhanced: if an object cannot be found on the FUSER file, the library SYSTEM on the system file FNAT is also searched for it. The search sequence is:

1. current "SYS" library (as defined by system variable *LIBRARY–ID),

2. steplibs (in sequence as specified in NATURAL SECURITY profile of current library),

3. library SYSTEM on FNAT system file,

4. library SYSTEM on FUSER system file (← *new!*).

Thus, it is possible to make user exits generally available without having to keep copies of them on both system files. It will be sufficient to provide them in one location, namely on the FUSER system file.

Size of Object Code

The maximum possible size of a (cataloged) NATURAL object module has been increased from 32 KB to 64 KB.

Size of Data Areas

The maximum possible size of a global data area or local data area has been increased from 32 KB to 16 MB.

Number of Variables in Programming Objects

Due to a restructuring of the NATURAL-internal symbol table, the number of variables per NATURAL programming object is in most cases no longer limited by the size of the symbol table.

Evaluation of Printer Definitions

With Version 2.2, a programming object cannot be compiled if no printer has been defined for the report number specified in the object (for example, a WRITE (5) statement cannot be compiled if printer 5 is not defined with the PRINTER profile parameter).

With Version 2.3, it is possible to compile a programming object even if the specified printer numbers have not yet been defined. The availability of the printers is checked at runtime.

Arithmetic Functions

With Version 2.3, NATURAL's internal handling of various arithmetic functions has been enhanced/corrected. This may in some cases lead to different results. However, the result will in all cases be more correct or of a greater precision respectively.

Computation of Floating-Point Exponentiation Corrected

With Version 2.2, if in an exponentiation both the base and the exponent are of floating-point format, the length of the *exponent* is used for the computation of the result.

With Version 2.3, if in an exponentiation both the base and the exponent are of floating-point format, the length of the *base* is used for the computation of the result.

With Version 2.2, if in an exponentiation the exponent is of floating-point format and the base is not, the base is internally converted to format/length F4 or F8, depending on the length of the base.

With Version 2.3, if in an exponentiation the exponent is of floating-point format and the base is not, the base is internally always converted to format/length F8 so as to get the greatest possible precision.

Both the above corrections may in some cases lead to different results; however, these results will be of a greater precision.

Results for SIN, COS and TAN Functions

With Version 2.2, when the mathematical functions SIN, COS and TAN (sine, cosine and tangent) are applied to very large numbers (equal to or greater than 10^{17}), they may in some cases return incorrect results.

With Version 2.3, for numbers equal to or greater than 10^{17} the sine will be "0", the cosine will be "1" and the tangent will be "0". This may in some cases lead to different results.

More Precise Results for SQRT Function

With Version 2.3, the computation of the mathematical function SQRT (square root) has been improved for floating-point operands.

This may in some cases lead to different results. However, these results will be of a greater precision.

Example:

```
SQRT(+1.735626284164890E+02)
```

Result with Version 2.2:   `+1.317431699999999E+01`

Result with Version 2.3:   `+1.317431700000000E+01`

Assignment of Numbers with Decimal Digits to Time Corrected

With Version 2.2, if numbers (format N or P) containing decimal positions are assigned/moved to a time field (format T) , the entire number is assigned/moved as an integer; that is, the decimal point is ignored.

With Version 2.3, this error has been corrected: The positions after the decimal point will be truncated, or rounded (if the ROUNDED option is used in the corresponding COMPUTE or MOVE statement). This may lead to different results — which will, however, be correct.

Example:

```
MOVE 1.23 TO #TIME-FIELD (T)
WRITE #TIME-FIELD (EM=HH:II:SS.T)
```

Result with Version 2.2:   `00:00:12.3`

Result with Version 2.3:   `00:00:00.1`

Assignment of Negative Numbers to Date and Time Intercepted

It is not allowed to assign a negative value to a date field (format D) or a time field (format T).

With Version 2.2, however, such invalid assignment at runtime may in some cases not be intercepted.

With Version 2.3, this has been corrected: The assignment of a negative value to a date or time field will always lead to an error (NAT1319).

More Precise Results for Arithmetic Expressions with Floating-Point Operands

If a floating-point operand is used in an arithmetic expression within a logical condition, the comparison is performed in floating point.

With Version 2.2, if the operators "=" or "NE" are used in such an arithmetic expression, the last hexadecimal mantissa digit is ignored to — supposedly — bypass rounding errors. However, due to the nature of floating-point numbers, rounding and truncation errors cannot be avoided when conversions to/from floating-point representation or computations with floating-point numbers are performed.

With Version 2.3, the above-mentioned "bypass" has been removed. This may in some cases lead to different results; however, these results will be of a greater precision.

More Precise Results for Floating-Point Conversions

The format conversion for the transfer of data from floating-point fields (format F) to packed numeric fields (format P) and vice versa, as well as from floating-point fields to alphanumeric fields (format A) and vice versa, has been improved. This may in some cases lead to different results. However, these results will be of a greater precision than with Version 2.2.

Example — Packed Numeric to Floating Point:

#PACKED (P29) contains: `12345678901234567890123456`

`MOVE #PACKED TO #FLOAT (F8)`

Result with Version 2.2: `12345678901234566365708288`

Result with Version 2.3: `12345678901234567707885568`

Example — Floating Point to Packed Numeric:

#FLOAT (F8) contains: `-19342813113834066526863360`

`MOVE #FLOAT TO #PACKED (P29)`

Result with Version 2.2: `-19342813113834066365802086`

Result with Version 2.3: `-19342813113834066526863360`

Example — Alphanumeric to Floating Point:

`COMPUTE #FIELD (F8) = VAL('1E9')`

Result with Version 2.2: `+9.999999999999992E+08`

Result with Version 2.3: `+1.000000000000000E+09`

Example — Floating Point to Alphanumeric:

#FLOAT (F8) contains: `0.000000099999999999999995474811182588...`

`MOVE #FLOAT TO #ALPHA (A20)`

Result with Version 2.2: `+9.999999999999997E-08`

Result with Version 2.3: `+1.000000000000000E-07`

Sign of Packed Numbers in Assignments

With Version 2.2, a positive sign is internally always represented as H'F'.

With Version 2.3, all valid positive signs are internally converted to either H'F' or H'C'; this can be controlled by an option of the new system command COMPOPT (see page 34).

With Version 2.2, an invalid sign may in some case assumed to be a negative sign.

With Version 2.3, an invalid sign is always assumed to be a positive sign. Moreover, the negative sign H'B' is converted to H'D'.

Assignments Between Numeric Variables of the Same Length

The internal handling of assignments of a value from one variable with format N to another variable with format N — where both variables have the same length — has been changed (so as to be consistent with the internal handling of assignments between format N variables of different lengths). When these variables are redefined, this may in some cases lead to different results.

A different result will also occur if the first three bytes of the content of the system variable *CURSOR represent a negative number and the content of *CURSOR is assigned to a user-defined variable of format/length N6.

Arithmetic Operations with NATURAL OPTIMIZER COMPILER

With Version 2.2, the use of the NATURAL OPTIMIZER COMPILER may in some arithmetic operations lead to different results than when performed without the NATURAL OPTIMIZER COMPILER.

With Version 2.3, all arithmetic operations always lead to the same results, whether the NATURAL OPTIMIZER COMPILER is used or not.

Array Operations with Variable Index Ranges

With Version 2.3, the handling at runtime of operations involving arrays with variable index ranges has been improved to avoid incorrect/inconsistent results. This affects the following:

Interception of Mismatching Array Ranges

With Version 2.2, some array operations using variable index ranges may lead to incorrect results if the array ranges do not match at runtime.

With Version 2.3, once the actual values are assigned to the variables in the index at runtime and the actual index ranges are thus determined, the actual range is compared with the syntax rules for *constant* index ranges: If the same construction using constant index ranges would lead to an error at compilation, an error will be issued at runtime (NAT1317).

Consequently, possible incorrect results that may have gone unnoticed with Version 2.2 will be intercepted by a runtime error with Version 2.3.

Comparison and Assignment of Variable Array Ranges

With Version 2.3, a comparison or assignment involving arrays with variable indexes will lead to an error at runtime (NAT1317) if an array range turns out to be actually a scalar once the actual values are assigned to the index variables.

With Version 2.2, such a comparison or assignment is allowed, but it is not consistent with the handling of constant scalars (as shown in the following example).

Example (assuming j = i + 1):

Version 2.2:

```
1.   IF #A(i:j) = #B(m)      is resolved as:  IF #A(i) = #B(m) OR  #A(j) = #B(m)

2.   IF #A(i:j) = #B(m:n)   is resolved as:  IF #A(i) = #B(m) AND #A(j) = #B(n)
```

This means that if the values of "m" and "n" are equal, comparison **2.** is resolved inconsistently.

Version 2.3:
If the values of "m" and "n" are equal, comparison **2.** will cause a runtime error.

Session Control

Terminate Session in Case of Initialization Error

With Version 2.2, a session initialization error does not prevent a NATURAL session from being continued: In online mode, the initialization errors are displayed and you can choose to either continue or terminate the session; in batch mode, the session is continued with the initialization errors going unnoticed — possibly leading to errors or undesired results later in the session.

With Version 2.3, the new profile parameter ITERM allows you to prevent a NATURAL session from being continued in the case of initialization errors. If an initialization error occurs, the session is then terminated immediately.

Suppress Command Mode

The new profile parameter CM allows you to suppress the NATURAL command mode (NEXT and MORE). As a result, the NATURAL session will be terminated whenever NEXT is encountered, and the MORE line will be write-protected (no input possible).

Disable Terminal Command "%%"

The new profile parameter ESCAPE allows you to disable the terminal command "%%". The terminal command will then be ignored; that is, it will not be possible to leave the currently active NATURAL program or the NATURAL session respectively by entering "%%".

Suppress Display of Session-End Message

By default, a message indicating that the NATURAL session has been ended normally (NAT9995) is displayed at the end of the session. The new profile parameter ENDMSG allows you to suppress this message.

Error Messages

Enhanced Message Texts

Descriptions of error messages have been extended to provide a more precise indication as to why an error condition has occurred.

Example:

Version 2.2:

```
NAT0082 Invalid command, or object does not exist in library.
```

Version 2.3:

```
NAT0082 object-type object-name not found in library.
```

```
NAT0082 Subprogram XYZ not found in library.
```

NAT1117 and NAT0924 Replaced by NAT0082

In situations where Version 2.2 displays error message NAT1117 (requested map not available) or NAT0924 (subroutine, GDA or external report not found), Version 2.3 displays message NAT0082. This will lead to different results if you interrogate these message numbers in your applications.

Avoid Truncation of Message Texts

By default, a NATURAL system error message consists of: the name of the program and the number of the line that caused the error, followed by the actual text of the message. Depending on the size of window in which the message is displayed, the actual text may be truncated.

To avoid this truncation, the new profile parameter MSGSF can be set so that only the actual message text itself — without program name and line number — will be displayed.

Database Access

Dynamically Loaded ADABAS Link Module

The new profile parameter ADANAME allows you to specify the name of the ADABAS link routine to be used. The specified link routine will then be loaded dynamically. Thus, it is no longer necessary to statically link the ADABAS link module to the NATURAL nucleus. Moreover, it is possible to run the same NATURAL nucleus with different ADABAS link modules without having to relink the NATURAL nucleus.

Reentrant ADABAS Link Routine

The ADABAS link routine can be assembled to be reentrant.

NATURAL always passes a reentrancy buffer with the 7th parameter.

Specify Database Types and Options Dynamically

The new profile parameter DB allows you to specify dynamically the same database specifications that can be specified with the NTDB macro in the parameter module.

Allow Database Open Despite Blank ETID

With Version 2.2, no database open is issued if the ETID is set to blanks.

With Version 2.3, the new profile parameter DBOPEN makes it possible to issue a database open despite a blank ETID.

Determine Number of Database Calls Before Roll-Out

With Version 2.3, the new profile parameter DBROLL determines the number of database calls after which a roll-out of the NATURAL thread will be performed. The NATURAL session is suspended during the roll-out.

The DBROLL parameter only applies under COM-PLETE and CICS.

Note: With Version 2.2, the number of database calls before roll-out is determined by the ADALIM parameter in the NTCOMP macro (for COM-PLETE) and the ADACALL parameter in the NCIPARM macro (for CICS) respectively.

Suppress BACKOUT TRANSACTION at Session End

With Version 2.2, an implicit BACKOUT TRANSACTION statement is automatically issued by NATURAL at the end of the NATURAL session.

With Version 2.3, the new profile parameter ENDBT allows you to determine whether an implicit BACKOUT TRANSACTION statement is to be issued at session end or not.

Control of END TRANSACTION at End of Program

With the new profile parameter ETEOP, you can determine whether an implicit END TRANSACTION statement is to be issued at the end of a NATURAL program (that is, before NEXT mode is reached) or not.

With Version 2.2, this functionality is controlled by the NOOPEN option of the OPRB profile parameter.

Issue END TRANSACTION upon Terminal I/O

With the new profile parameter ETIO, you can cause NATURAL to issue an implicit END TRANSACTION statement whenever a terminal I/O occurs. Whenever a transaction monitor commits the associated databases because of a terminal I/O, all related databases are also committed. This is useful for the synchronization of database transactions.

Ignore Response Code 113 for FIND and GET

By default, ADABAS response code 113 (requested ISN not found) causes the program to be terminated. The new profile parameters RCFIND and RCGET allow you to cause processing to continue in the case of response code 113:

- RCFIND causes processing to continue in the case of response code 113 being returned during the execution of a FIND loop. Response code 113 will then be ignored, and processing of the FIND loop will be continued by reading the next record.

- RCGET causes processing to continue in the case of response code 113 being returned during the execution of a GET statement. Response code 113 will then be ignored, the system variable *ISN will be set to "0", and processing will be continued.

Terminal I/O Handling

Repeat Screen After CALL (%RN)

At a screen I/O, NATURAL, by default, does not send the entire screen data to the screen but only those parts of the screen that have changed since the previous screen I/O. However, when a non-NATURAL program is invoked (with a CALL statement) which issues screen I/Os, the screen changes caused by the non-NATURAL program will (under some TP monitors) go unnoticed by NATURAL — consequently, the next screen I/O after the CALL may cause the screen data to get mixed up.

With Version 2.3, the new terminal command %RN allows you to avoid such a mix-up: %RN suppresses NATURAL's screen optimization for the next screen I/O — which means that the entire screen data is sent (instead of only the changed parts).

Suppress Internal REINPUT for Invalid Data

By default, NATURAL automatically issues an internal REINPUT statement if invalid data have been entered. With the new profile parameter REINP, you can switch this mechanism off. This allows you to handle such input errors yourself in your application.

Set Terminal Type at Session Initialization

The new profile parameter TTYPE has the same function as the terminal command "%T=": it allows you to specify the terminal type used (in TP environments in which this information is not supplied automatically), so that NATURAL can activate the appropriate converter routine for attribute sequences to operate that type of terminal.

With the TTYPE parameter, you can set the terminal type when you invoke NATURAL. This means that it is no longer necessary to execute a program containing a SET CONTROL 'T=...' statement at the start of the session in order to set the terminal type.

Switch Off Compression of Screen Data (for 3270-Type Terminals)

With the new profile parameter DSC, it is possible to switch off NATURAL's automatic optimization of the screen data stream for 3270-type terminals.

NATURAL's screen optimization causes screen data to be sent as compressed as possible. If this should conflict with any TP monitor's screen optimization or hardware limitiation, you can use this parameter to switch off NATURAL's screen optimization; screen data will then be sent in non-compressed form.

Note: The DSC parameter has the same function as the terminal command %RO.

Handling of Protected Fields

Suppress Overwriting of Protected Fields by Helproutines

With Version 2.2, a helproutine assigned to a field can overwrite the field's content, even if the field is write-protected (AD=P).

With Version 2.3, the new profile parameter OPF allows you to suppress this, so that helproutines cannot overwrite the contents of write-protected fields.

Suppress Filler Character for Dynamically Protected Input Fields

The new profile parameter FCDP allows you to suppress the display of filler characters for input fields that have been made write-protected dynamically (that is, to which the attribute AD=P has been assigned via a control variable).

With Version 2.2, a dynamically protected input field is displayed filled with filler characters — which may suggest to the users that they could enter something in the field.

With Version 2.3, you can avoid this by setting FCDP=OFF: Dynamically protected input fields will then be displayed filled with blanks instead of filler characters.

Suppress Zero Display for Time Fields

With Version 2.2, the profile/session parameter ZP, which can be used to suppress the display of a field values that consist of all zeros, applies only to numeric fields (formats N, P, I and F).

With Version 2.3, the ZP parameter also applies to time fields (format T).

Loading of Datasets with INPL

With Version 2.3, the loading with INPL of datasets into the system files is restricted to datasets that are identified as official SOFTWARE AG INPL system datasets.

In addition, objects in INPL datasets to be loaded into library SYSTEM are no longer loaded into the system files FNAT and FUSER, but only into the FNAT system file.

Support of Kanji Printing

With Version 2.3, NATURAL supports the printing of Kanji output. This is controlled via the new macro NTKAPRI in the NATURAL parameter module, and the corresponding new profile parameter KAPRI.

Improved Handling of Internal Buffers

With Version 2.2, the profile parameter AVERIO controls the sizes of various internal NATURAL buffers: the page attribute buffer, the overlay attribute buffer and the screen attribute buffer.

With Version 2.3, NATURAL automatically adjusts the sizes of these buffers during the session according to the actual requirements. Thus, buffer overflow errors are avoided. The AVERIO parameter is obsolete.

SOFTWARE AG Editor

As the RECOVER mechanism of SOFTWARE AG Editor Version 2.2.6 (or above) is upward compatible, you can continue to use a Version 2.2 editor buffer pool work file for Version 2.3.

However, you cannot use the same editor work file for both Versions 2.2 and 2.3 at the same time.

If you wish to use both a Version 2.2 editor work file and a Version 2.3 editor work file concurrently in the same region, you must use different logical dataset names (DDNAMEs) for these work files.

Under COM-PLETE, a VSAM dataset is no longer used as editor work file. Therefore, the editor buffer pool under COM-PLETE cannot be shared with other environments.

Dump Analysis Tool SYSNDA

The SYSNDA utility, which has been available on request, is no longer available as of Version 2.3.

NATURAL OPTIMIZER COMPILER

The following enhancements are provided with NATURAL Version 2.3:

- It is possible to apply the NATURAL debugging facility to programs compiled with the NATURAL OPTIMIZER COMPILER.

- For arithmetic operations with the NATURAL OPTIMIZER COMPILER, see page 99.

For an overview of the enhancements to the NATURAL OPTIMIZER COMPILER itself, please refer to the *NATURAL OPTIMIZER COMPILER Manual*.

ENTIRE Connection

The following enhancements are provided with NATURAL Version 2.3:

- With ENTIRE Connection Version 2.1.1 and above, the terminal models 3, 4 and 5 are supported.

- It is possible to download time and date variables from NATURAL programs.

As of NATURAL Version 2.3, support of NET-PASS Asynchronous (Combord NPA) is only available on request.

NATURAL SECURITY

This chapter contains information on:

- support of mainframe as server,
- no migration/conversion required,
- logging of maintenance functions,
- enhanced password protection,
- new mechanism to protect NATURAL utilities,
- user maintenance enhancements,
- library maintenance enhancements,
- revised handling of DDMs,
- logon records,
- logon/countersign error processing,
- suppress display of logon messages,
- SECLOAD enhancements,
- invoking user-written programs with direct commands,
- functional security,
- mailboxes,
- administrator services menu,
- subprogram SECQ removed.

Support of Mainframe as Server

At present, in a client/server environment only UNIX can be used as a NATURAL SECURITY server.

With Version 2.3, NATURAL SECURITY on a mainframe computer can also be used as server for any clients.

No Migration/Conversion Required

With NATURAL SECURITY Version 2.3, you can continue to use an existing Version 2.2 FSEC system file; that is, it is not necessary to transfer existing Version 2.2 security data to a new FSEC file. Also, you can continue to use your Version 2.2. security data without having to convert them.

Logging of Maintenance Functions

A new logging function for maintenance activities within NATURAL SECURITY allows you to ascertain who has last made which modifications to which security profile and to which administrator services setting.

To activate the logging function and determine which modifications are to be logged, you use the new general option "Logging of maintenance functions" in Administrator Services.

To view the log records that are written when the logging function is active, you use the new Administrator Services function "Processing of maintenance log records".

Logon Monitoring

The function "Processing of maintenance log records" also provides an option which allows you to specify a time interval and ascertain which users have not logged on within the last *n* days. This option also allows you to ascertain when each user has logged on last.

Enhanced Password Protection

In addition to the already existing password options (minimum length, expiration date), the new Administrator Services functions "User Password History" and "Password Rules" are provided, which allow you to enforce a more efficient password protection. The following options are available with these functions:

- The password must not be identical to the user ID (as contained in the NATURAL system variable *USER) or part thereof.

- The last *n* passwords used by the user cannot be used again as new password.

- A new password must not be too similar to the old one (this check will reject a new password if its last three characters are identical to those of the old password).

- Passwords must conform to "masks"; that is you can define rules to which each character in a password must adhere (for example, the first 3 characters must be letters, or the last character must not be a number).

- Passwords must not contain double characters (e.g. the T's in "LITTLE").

- Passwords must not contain a character twice (e.g. the T's in "THIRST").

- With Version 2.2, when a newly defined user logs on for the first time and the password is identical to the user ID, the user must change his/her password at the first logon.
  With Version 2.3, a user must also change his/her password at logon whenever the password has been changed by a NATURAL SECURITY administrator in the user security profile.

New Mechanism to Protect NATURAL Utilities

NATURAL SECURITY provides a new mechanism to control the use of NATURAL utilities.

Note: The old utility protection mechanism is still available with Version 2.3; however, it will be removed with one of the next versions of NATURAL SECURITY. In any case, it is strongly recommended that the new mechanism be used, as it provides for a much more sensible and efficient protection of utility functions.

The new mechanism can be applied to the utilities NATLOAD, NATUNLD, SYSBPM, SYSDDM, SYSERR, SYSMAIN, SYSPARM and SYSTRANS.

The following provides a brief overview of the concept of the new mechanism. For details, see the chapter **Controlling the Use of NATURAL Utilities** in the *NATURAL SECURITY Manual for Mainframes*.

The new utility protection is function-oriented, which means that it is based on the concept that you can allow or disallow individual functions of a utility. To control the use of a utility, you need not define a library security profile for it. Instead, you control it by defining *utility profiles* for it. In a utility profile, you allow and disallow the individual functions of the utility. Basically, a utility profile consists of a list of the utility's functions, each of which can allow or disallow by marking it with "A" or "D" respectively.

For each utility, you can define:

*   a default profile,
*   user-specific profiles,
*   library-specific profiles,
*   user-library-specific profiles.

Default Utility Profile

The *default profile* of a utility applies to all users (except those for which user-specific profiles are defined). It determines which of the utility's functions the users may use and which not.

User-Specific Utility Profiles

If an individual user is to use (or not to use) other functions than the other users, you can define a *user-specific utility profile*, which overrides the default profile and determines which of the utility's functions this particular user may use and which not.

Library-Specific Utility Profiles

Several utilities affect individual NATURAL libraries (for example, SYSERR can be used to maintain error messages that belong to a specific library). Generally, the utility's default profile applies to all affected libraries.

However, if some of the utility's functions are only to be allowed/disallowed for a particular library, you can define a *library-specific utility profile*, which overrides the default profile as well as any user-specific profiles for that utility, and determines which of the utility's functions may be applied to this particular library and which not.

User-Library-Specific Utility Profiles

For utilities which affect individual NATURAL libraries, two kinds of situations may occur:

- A *user-specific* utility profile determines which of a utility's functions a particular user may use, regardless of the libraries which are affected by the functions (provided that no *library-specific* profiles are defined for this utility). However, if this user is to have different functions usage permissions for a particular library affected by the utility's functions, you can define these in a *user-library-specific* utility profile.

- A *library-specific* utility profile determines which of a utility's functions may be used when applied to a particular library; for this library, it applies for all users (regardless of any *user-specific* profiles). However, if a particular user is to have different functions usage permissions for this library, you can define these in a *user-library-specific* utility profile.

A *user-library-specific utility profile* only applies for one user and one library, it overrides any other profiles for this library and this user, and it determines which of the utility's functions this user may use for this library.

How to Define Profiles

To define default profiles, you use the new Administrator Services function "Definition of utility defaults/templates". To define all other types of utility profiles, you use the functions provided by the new Utility Maintenance subsystem.

Retrieval

New retrieval options allow you to ascertain which utility profiles apply for which utility, which user and which library.

User Maintenance Enhancements

Renaming of User Security Profiles

With Version 2.2, the only way to rename a user security profile is to copy the profile to another ID and then delete the old profile.

With Version 2.3, a rename function (function code "RE" on the User Maintenance selection list) is provided, which allows you to rename a user security profile by simply changing its ID. Any links etc. will be automatically adjusted.

Activation Dates

To comply with year 2000 requirements, the activation dates of user security profiles can now be specified with a 4-digit year component.

Private Library

The Private Library section of the Additional Options of a user security profile can now be invoked directly from the main user security profile screen by pressing PF5.

Library Maintenance Enhancements

Set Status of DDMs

This new option allows you to use DDMs which are not to be protected, without having to define file security profiles for them.

This option can be set in the Restrictions part of a library security profile. It only affects DDMs for which no file security profiles have been defined, and it only applies if "Transition Period Logon" is set to "N".

By default, this option is set to "UNDF", which means that DDMs for which no file security profiles have been defined cannot be used by any program in this library.

If you set this option to "PUBL", the status of all DDMs for which no file security profiles have been defined is assumed to be PUBLIC, which means that these DDMs can be used by any program in this library. This allows you to use these DDMs without having to define file security profiles for them.

For all other DDMs, their respective file security profiles determine if and from which libraries they can be used.

Renaming of Library Security Profiles

With Version 2.2, the only way to rename a library security profile is to copy the profile to another ID and then delete the old profile.

With Version 2.3, a rename function (function code "RE" on the Library Maintenance selection list) is provided, which allows you to rename a library security profile by simply changing its ID. Any links etc. will be automatically adjusted.

Maximum Number of ADABAS Calls

With Version 2.2, if you wish no limit to be in effect for the maximum number of ADABAS calls permitted between two screen I/O operations, you have to set to "0" both the field "Maximum Number of ADABAS Calls" in a library security profile as well as the corresponding NATURAL profile parameter MADIO.

With Version 2.3, if you wish no limit to be in effect, it is sufficient to set the field in the library profile to "99999" (any setting of the MADIO parameter will then be ignored).

NATURAL RPC Restrictions

When you press PF8 on the Session Parameters screen in the Restrictions section of a library security profile, another screen will be displayed in which you can set various restrictions that apply when subprograms contained in the library are executed via NATURAL RPC in a client/server environment. For NATURAL RPC under NATURAL SECURITY, see also page 91.

Cross-Reference

With the Cross-Reference option in library security profiles, you can specify the new value "D", which corresponds to the new value "DOC" of the system command XREF (see page 40).

Restrictions

The Restrictions section of the Additional Options of a library security profile can now be invoked directly from the main library security profile screen by pressing PF5.

Steplibs

With Version 2.2, if you assign "999" as DBID value for a steplib in the steplib window of a library profile, the DBID value specified in the library profile of the steplib will be used.

With Version 2.3, you have to specify "99999" instead of "999" for this purpose. However, the old value of "999" is still valid for compatibility reasons.

Revised Handling of DDMs

Contrary to the announcement made in the *NATURAL Version 2.3 Planning Guide for Mainframes*, NATURAL SECURITY's handling of DDMs has *not* been revised entirely. In other words, the existing "File Maintenance" functionality continues to be available unchanged.

The only enhancement to the handling of DDMs is the new option "Set Status of DDMs", which can be set in library security profiles (see above).

Logon Records

User-Specific Writing of Logon Records

With Version 2.2, the writing of logon records is controlled by the option "Logon recorded" in *library* security profiles so that a logon record is written every time any user logs on the library, which means that logons can only be monitored on a library-by-library basis.

With Version 2.3, the option "Logon recorded" can also be set in *user* security profiles. A logon record will then be written every time the user logs on to any library. This allows you to monitor logons of individual users.

Listing Logon Records of Undefined Users/Libraries

If the general option "Transition Period Logon" is set to "Y", a logon record is also written every time an undefined user logs on, and every time a user logs on to an undefined library.

With Version 2.2, these records cannot be listed separately.

With Version 2.3, the new selection criteria "UX" and "LX" can be specified on the Logon Records Processing Menu; they allow you to list only logon records of undefined users and only logon records to undefined libraries respectively.

Deletion of Logon Records

With Version 2.2, logon records can only be deleted page by page.

With Version 2.3, the new direct command LOGDEL allows you to delete *all* logon records at once.

Logon/Countersign Error Processing

Display of Individual Error Records

The display of individual error entries ("error history") has been enhanced, including the date and time when the error was produced.

Deletion of Error Entries

With Version 2.2, logon/countersign error entries can only be deleted individually or page by page.

With Version 2.3, the new direct command ERRDEL allows you to delete *all* logon/countersign error entries at once.

Suppress Display of Logon Messages

A new general option in Administrator Services allows you to suppress the display of the messages NAT0853 and NAT0854, which indicate that a logon to a library has been successful. By default, one of these messages is displayed after every successful logon to a library.

SECLOAD Enhancements

The program SECLOAD, which is used to load NATURAL SECURITY data to another system file and/or to another hardware platform, has been enhanced to allow you to transfer data to another NATURAL SECURITY version and/or to another platform. Thus, you can, for example, transfer data from Version 2.2 on UNIX to Version 2.3 on a mainframe computer.

For the transfer between Versions 2.2 and 2.3, SECLOAD automatically performs the necessary version-dependent conversion of the transferred data. This means the conversion program CONVERT is no longer required (except for the conversion of Version 2.1 data to Version 2.2 or 2.3).

Invoking User-Written Programs with Direct Commands

The new direct commands CUSTOM1, CUSTOM2, CUSTOM3, CUSTOM4 and CUSTOM5 invoke NATURAL programs of the same names. You can write your own programs of these names to perform whatever functions you require. This allows you to invoke such functions from within NATURAL SECURITY.

Functional Security

Status of a Command Processor

Since NATURAL SECURITY Version 2.2.8, the status of a command processor can be:

Undefined	The command processor has been created with SYSNCP, but no functional security is defined for it.
Defined	The command processor has been created with SYSNCP and functional security is defined for it.
Modified	The command processor has been modified with SYSNCP after functional security was defined for it. In this case, you may have to update the functional security for the command processor (by marking the field "Functional Security Defined" with "UP" and then adjusting the security specifications).
Unresolved	The command processor has been deleted with SYSNCP, but functional security is still defined for it.

The status "Modified", however, had not been implemented with Version 2.2.8. It has been implemented with Version 2.3.

Mailboxes

Expiration Date

To comply with year 2000 requirements, the expiration date of a mailbox can now be specified with a 4-digit year component.

Administrator Services Menu

Due to the increased number of Administrator Services functions, the Administrator Services Menu now consists of two screens. With PF7 and PF8, you can switch between the two screens.

Subprogram SECQ Removed

As of Version 2.3, the NATURAL SECURITY subprogram SECQ, which was available on request with Version 2.1 and is reserved for internal use, is no longer available.

Instead of SECQ, the — already existing — interface subprogram NSCXR should be used.

NATURAL ADVANCED FACILITIES

This chapter contains information on:

- new spool file layout,
- new user interface,
- owners,
- statistics,
- logging,
- notes,
- applications,
- default object definitions,
- mass updates,
- printout sequence,
- user profiles,
- logical printers,
- physical printers,
- allocation,
- spool file administration.

New Spool File Layout

With Version 2.3, the layout of the spool file has been changed, using a new FDT (field description table). A conversion routine is provided which allows you to convert objects from a Version 2.2 spool file to a new Version 2.3 spool file.

Note: Report data cannot be converted to the new spool file layout. This means that you have to print them before you run the conversion routine.

New User Interface

The user interface of NATSPOOL has been revised to make navigation within NATSPOOL and the execution of functions easier, and to display more information on objects.

Some of the major new features of the new user interface are:

- **Command Line** — Every menu screen provides a command line. Apart from pressing a PF-key or using cursor selection, you can invoke a function via a direct command you enter in the command line.

- **Device Administration** — The display of device attributes has been enhanced, and additional information on devices is displayed. Moreover, several new line commands for device administration are available.

- **Report / Queue Administration** — The display of report attributes has been enhanced, and additional information on reports is displayed. Also, if more than one printer is active for a queue (destination/form), this information is displayed. Moreover, several new line commands for report / queue administration are available.

- **CALLNAT Interface** — The CALLNAT interface, that is, the subprograms delivered with NATURAL ADVANCED FACILITIES, are documented and directly accessible online: Example programs of how to invoke the subprograms are provided online, and you can view, modify and execute the examples by selecting them from a menu. Several new subprograms are also provided.

- **Session Information** — This new function shows the currently assigned spool file (database ID, file number), the current status of the spool options and the currently active hardcopy device. The hardcopy device entry is modifiable.

Owners

For each object defined on the spool file, you can specify user IDs as owners. Only these owners are then able to maintain the object. Thus you can restrict the maintenance of objects to specific users.

If NATURAL SECURITY is installed, you can also specify NATURAL SECURITY group IDs, thus allowing all users contained in a group to maintain the respective object.

Statistics

A new statistics function allows you to obtain information about the number of lines, pages and reports printed. You can get statistical information for a specific physical printer, user ID or allocation.

It is also possible to limit the activation of the statistics to a specified time window.

You can display, print and reset the statistics information as well as sort them by various criteria (for example, number of lines or pages per report, number of reports per printer).

Logging

A logging function is provided for objects on the spool file and for reports.

The logging functions logs the last 12 modifications of the object/report. The information logged for each modification includes the date, time, user ID and executed function.

Notes

For each NATSPOOL object, a text field (up to 64 bytes) is provided, in which you can enter your notes or comments concerning the object.

Applications

It is possible to define applications. For each application, you can specify a library name, the name of startup program (optional) as well as a descriptive application name (to be displayed on the applications selection list in NATSPOOL).

Once defined, an application can be invoked via the NATSPOOL main menu. This allows you to invoke another NATURAL application from within SYSPOOL, and afterwards return to SYSPOOL.

Moreover, you can restrict the invoking of an application to individual users.

Default Object Definitions

For ease of maintenance, it is possible to create default object definitions for the different object types. When you add a new object to the spool file, you can then use the predefined default object as the basis for the object you are creating (instead of your having to make all specifications by hand). The specifications of the predefined default object are then automatically used for the object you are creating.

Mass Updates

You can make modifications to all objects of type Logical Printer, Allocation and Physical Printer simultaneously.

You select the field(s) to be modified, then you specify the new value(s) for the field(s), then you apply the modification.

You can apply the modifications with or without individual confirmation.

Printout Sequence

With Version 2.2, reports with identical print criteria are printed in the sequence of the ISNs under which they are stored on the spool file.

With Version 2.3, reports with identical print criteria are printed in the sequence of their creation dates/times; that is, those created first are printed first.

User Profiles

Default Hardcopy Printer

In the user profiles, you can specify an additional logical printer which will be used as default hardcopy printer for the user.

Logical Printers

Online NTCC Maintenance

With Version 2.2, printer control sequences have to be specified in the macro NTCC, which is part of the configuration module NATCONFG linked with the NATURAL nucleus.

With Version 2.3, you can set and modify the NTCC specifications online in NATSPOOL.

The profiles defined in the NTCC macro are assigned to specific reports, regardless of the printer on which the reports are printed. The NTCC profiles defined in NATSPOOL, however, are printer-specific; that is, for the preparation of the reports, the printer type used will be taken into account and the printer control sequences set accordingly.

In the NATSPOOL NTCC profiles, it is possible not only to define printer control characters for the default attributes supported by NATURAL (underlined, boldface, etc.), but also control characters for additional special attributes you can define yourself.

A user attribute is defined by a text information, which will be replaced by the printer-specific control sequence.

Retention Period / Calendars

For each report a time interval (a number of days) can be defined after which the report will be deleted. This period can be set for each kind of disposition.

The time interval can be either an absolute number of days, counting every day; or it can be set so as not to count holidays, weekends etc. For the latter option, you will be able to define specific calendars.

In a calendar, which can span up to 24 years, days can be marked as working days or holidays, and weekends and the first day of the week can be set.

Printer Type

A logical printer can be defined for different spooling systems. It is possible to route reports directly to ENTIRE OUTPUT MANAGEMENT or to the Siemens RSO spool.

Moreover, it is possible to specify spooling-system-specific parameters.

Clusters

When using different environments (computer centers, distributed locations, etc.), you can define clusters of logical printers. A cluster is a set of logical printers.

Thus, it is possible to restrict the maintenance of logical printers (and corresponding allocations and physical printers) to specific clusters.

A logical printer cannot be included in more than one cluster.

Default Logical Printer

It is possible to define a default logical printer. This printer will be used for a report number to which no logical printer has been assigned in NATSPOOL, or to which a non-existing printer name has been assigned in a DEFINE PRINTER statement.

Report Protection

You can protect a report against being read, deleted and/or printed.

Physical Printers

Number of Users per Private Printer

With Version 2.2, the maximum number of user IDs that can be assigned to a "private" printer is 30.

With Version 2.3, the number of user IDs per "private" printer is not limited.

Operating System / TP Monitor

When using NATSPOOL in heterogeneous environments, you can specify the operating system/TP monitor for which a printer definition is to be used (BS2000, CICS, IMS/TM).

Form Check

It is possible to activate a check which compares the current form with the one printed last. If they differ, the current form cannot be printed. For the check, an initial value (destination/form) can be set.

BS2000-Specific Enhancements

Under BS2000, the following NATSPOOL enhancements are available:

- A trace function is provided to get information about the execution of all DCAM calls used by the spool server.

- It is possible to activate a restart function and also to restrict the number of restart attempts for a printer with status INOP (1 to 98 = number of restarts, 99 = unlimited, 0 = no restart).

- For the RSO spool, a default form name can be defined.

- NATURAL ADVANCED FACILITIES provides three exits — NAFEXIT1, NAFEXIT2 and NAFEXIT3 — which support user-written Assembler routines. The passing of parameters to and from these exits has been changed to conform to industry standard conventions.

Allocation

Header Pages

You can define header pages (of up to 62 lines and 130 columns) and assign them to reports. A header page is then always printed immediately before a report to which it is assigned. Within a header page, you can specify variables which will be replaced by corresponding current values (for example, date, time, user ID) when a report is printed.

Hardcopy Allocation

It is possible to make the allocation of hardcopy printers dependent on either user IDs or terminal IDs: The logical printer whose name corresponds to the user's user ID or terminal ID respectively will then be used as hardcopy printer.

In addition, it is possible to set positions in the user/terminal IDs which are not to be evaluated. For example, if only the first three characters are to be evaluated, all IDs with the same first three characters will use the same logical printer (e.g., the user/terminal IDs "ABC1", "ABC22" and "ABCDEF" will use the logical printer "ABC———"). Thus, multiple users/terminals can share the same hardcopy printer.

Queues

All reports assigned to an allocation are called queue. It is possible to deactivate a queue to prevent it from being started (either automatically or by NATSPOOL administration).

When using NATSPOOL under BS2000, you can define a queue as "monitor queue": a monitor task will then schedule the printing of reports which are created for this queue.

Backup and Alternate Printers

For physical printers 2 – 16, you can make one of the following type specifications:
* **Backup**   — The printer is only used when the other printers are inoperable.
* **Alternate** — The printer is used when the other printers are already in use.

Time Windows (under BS2000 only)

When using NATSPOOL under BS2000, you can define time windows for an automatic start of reports.

Spool File Administration

Layout of Spool File

This function provides the following enhancements:

- It is possible to modify the spool file options without having to newly format the spool file for the modifications to take effect.

- It is possible to change the password without having to newly format the spool file for the change to take effect.

- It is possible to reset the password to the original default password (if the current password cannot be remembered).

- It is possible to display information (date, time, user ID, function performed) on the last modification of the spool file and password.

Check Spool File

This function provides the following enhancements:

- It is possible to check the relationship of all objects for consistency — and, if applicable, delete unused objects.

- It is possible to check if a logical printer is not contained in more than one cluster.

- It is possible to modify the synchronization flags concerning the restart of reports, time windows and delete functions of the spool server(s).

- For BS2000, it is possible to check the status of the defined DCAM applications.

Notes